

BELLCOMM, INC.

955 L'ENFANT PLAZA NORTH, S.W., WASHINGTON, 20024

COVER SHEET FOR TECHNICAL MEMORANDUM

TITLE- PAP - Parametric Analysis Program

TM-70-1032-1

FILING CASE NO(S)- 804

DATE- January 21, 1970

FILING SUBJECT(S)- Parameter Studies
(ASSIGNED BY AUTHOR(S))- Systems of Nonlinear Functions

AUTHOR(S)- P. F. Long

ABSTRACT

PAP provides a systematic approach to Parametric Analysis of systems whose characteristics (dependent variables) are continuous functions of the system parameters (independent variables). It is designed to explore the parameter space of a system using as few system evaluations as possible. This is achieved by:

1. Program design features which reduce the number of approaches to the computer required to obtain a given degree of coverage of the parameter space.
2. Algorithms specifically designed to minimize the number of system evaluations.
3. Extensive use of interpolation and extrapolation.
4. The provision of a flexible contour plotting capability..

The approach taken by PAP is to apply $n-2$ (n is the number of system parameters) constraints to the parameter space for fixed values of the remaining two parameters. Determine solutions to this nonlinear constraint problem and map the solutions found to the points of a grid determined by the domains of the two parameters previously fixed. This gives rise to a number of functions of two variables defined at discrete points which represent the behavior of the constrained system. PAP provides an interpolation scheme to generate contours of these functions and a flexible plotting capability so that they may be displayed in various combinations.

FACILITY FORM 802	N70-27160		(THRU)
	(ACCESSION NUMBER)		
	93		(CODE)
	(PAGES)		
CA-109832		19	(CATEGORY)
(NASA CR OR TMX OR AD NUMBER)			

SEE REVERSE SIDE FOR DISTRIBUTION LIST

Reproduced by
**NATIONAL TECHNICAL
 INFORMATION SERVICE**
 Springfield, Va. 22151

DISTRIBUTION

COMPLETE MEMORANDUM TO

CORRESPONDENCE FILES:

OFFICIAL FILE COPY

plus one white copy for each
additional case referenced

TECHNICAL LIBRARY (4)

Bellcomm, Inc.

R. A. Bass
G. C. Bill
A. P. Boysen, Jr.
F. A. Brewer
B. J. Brooks
C. L. Greer
R. W. Grutzner
D. R. Hagner
B. T. Howard
R. F. Jessup
M. G. Kelly
M. R. Kerr
D. L. Mather
J. Z. Menard
J. M. Nervik
M. P. Odle
J. R. Pascher
I. M. Ross
F. N. Schmidt
R. R. Singers
J. W. Timko
R. A. Troester
R. L. Wagner
P. A. Whitlock
M. P. Wilson
G. D. Wolske

Computer Library (4)

COVER SHEET ONLY TO

LESS APPENDIX C TO

G. R. Andersen
G. M. Anderson
D. R. Anselmo
J. O. Cappellari, Jr.
K. R. Carpenter
D. E. Cassidy
D. A. Corey
C. L. Davis
D. A. De Graaf
J. P. Downs
W. H. Eilertson
C. O. Guffee
W. G. Heffron
H. A. Helm
R. B. Hoekstra
J. Kranton
B. W. Lab
S. L. Levie
H. S. London
D. Macchia
E. D. Marion
K. E. Martersteck
H. H. McAdams
B. G. Niedfeldt
J. A. Schelke
J. J. Schoch
A. L. Schreiber
P. G. Smith
A. A. VanderVeen
J. E. Waldo

BELLCOMM, INC.

955 L'ENFANT PLAZA NORTH, S.W. WASHINGTON, D. C. 20024

SUBJECT: PAP - Parametric Analysis Program
Case 804

DATE: January 21, 1970

FROM: P. F. Long

TM-70-1032-1

TECHNICAL MEMORANDUM

INTRODUCTION

The nature of parametric analysis is exploratory. A number of different systems must be studied to determine the one most likely to accomplish the required goals. Each system usually involves a number of parameters (independent variables) which determine the characteristics (dependent variables) of that system. Parametric analysis involves the study of the various characteristics of a system as functions of its parameters. By getting an overview of how the system characteristics behave as the parameters vary over their permissible values, the most promising systems can be selected for more careful study. At the single system level, this overview permits the determination of subsets of the permissible values of the parameters for which the characteristics are favorable. More careful attention can then be given to these subsets.

A few standard mathematical notions necessary for the following discussion will be reviewed here. Standard set notation is used, e.g., $\{x: 1 \leq x \leq 2\}$ is the set of all real numbers between and including 1 and 2, alternately denoted by $[1,2]$. The notation (a,b) is used to denote the ordered pair a,b . Let A and B be arbitrary sets, then $A \times B$ is defined to be the set $\{(a,b): a \in A, b \in B\}$, where " \in " means "is an element of". More generally, if D_1, D_2, \dots, D_k are arbitrary sets, then $D_1 \times D_2 \times \dots \times D_k = \{(d_1, d_2, \dots, d_k): d_i \in D_i, i=1, \dots, k\}$. Let D and R be arbitrary sets, the set $\{(x,y): x \in D, y \in R\} = F$ is a function if and only if $(x,y_1) \in F$ and $(x,y_2) \in F$ implies that $y_1=y_2$. D is called the domain of the function and R its range. Note that D and R are arbitrary sets; in particular, they may be sets of n and m dimensional vectors respectively. In which case, the function associates with each vector in D a unique vector in R .

Parametric analysis is equivalent to the study of m functions, c_1, c_2, \dots, c_m , (the system characteristics) of n variables, p_1, p_2, \dots, p_n , (the system parameters). Using vector notation let $C = (c_1, c_2, \dots, c_m)$ and $P = (p_1, p_2, \dots, p_n)$, then parametric analysis is the study of the function $\{(P, C(P)) : P \in D\}$ where $D = D_1 \times D_2 \times \dots \times D_n$ and D_i is the domain of p_i , $i = 1, 2, \dots, n$. D is called the parameter space of the system. The objective of parametric analysis is to determine regions of the parameter space in which the characteristics are suitable. More precise simulation and/or more careful study can then be used to narrow these regions even further. In the final analysis, a single point of the parameter space will be chosen which represents what is hoped to be the best system for the required purpose.

Parametric analysis is often performed as follows: Suppose that n parameters are available, equality constraints are imposed on $n-1$ of the characteristics (for convenience say c_1, c_2, \dots, c_{n-1}), i.e., for a fixed value of one of the parameters (say p_1) p_2, p_3, \dots, p_n are determined such that $c_i = k_i$, $i = 1, 2, \dots, n-1$. This process is repeated for different values of p_1 , until its domain is adequately covered. Various variables are then plotted as functions of p_1 . This memorandum describes a method of parametric analysis (and the computer program which implements it) which substantially improves upon this procedure.

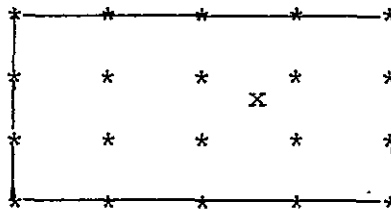
Functional Description

PAP uses a general technique for parametric analysis that is independent of, but easily adaptable to, particular studies. A number of procedures (PROC's) supplied by the user adapt PAP to the user's application. The system under study must be simulated by a number of user supplied subroutines. These subroutines and procedures are defined in detail later.

Consider a system described by the characteristics $C = (c_1, c_2, \dots, c_m)$ which are continuous functions of the parameters $P = (p_1, p_2, \dots, p_n)$. Let the interval $D_i = [a_i, b_i]$ be the domain of p_i , $i = 1, 2, \dots, n$. Suppose that for each p_i , a Δp_i is given which divides the interval $[a_i, b_i]$ into an

integral number of subintervals, i.e., $(b_i - a_i)/\Delta p_i = q_i$ is an integer, $i = 1, 2, \dots, n$. Define the sets $Q_i = \{v_j: v_j = a_i + (j-1)\Delta p_i, j = 1, 2, \dots, q_i+1\}$, $i = 1, 2, \dots, n$. PAP provides four major functions and an interface capability:

1. For a fixed point of $D_1 \times D_2$, an attempt is made to find all points of $D_3 \times D_4 \times \dots \times D_n$ for which $n-2$ of the characteristics have specified values, i.e., for which $c_i = k_i$, $i = 1, 2, \dots, n-2$. This offers a number of advantages over the practice of making one attempt per run to find one solution:
 - a. Potentially all solutions will be found.
 - b. When none are found, it is strong evidence that no solutions exist for the given constraints in the particular parameter space under study.
 - c. Given that complete coverage of the parameter space is required, the method provided tends to minimize the number of system evaluations required.
2. Consider the diagram below, $D_1 \times D_2$ is represented by the rectangle; $Q_1 \times Q_2$ is represented by the '*'s; and the fixed point (p_1, p_2) is represented by x. An attempt is made to map each solution found at (p_1, p_2) to all points of $Q_1 \times Q_2$. This means that each solution found for the fixed point $(p_1, p_2) \in D_1 \times D_2$ is used as the initial value in the iterative attempt to determine the corresponding solution at some or all of its four nearest neighbors in $Q_1 \times Q_2$.



Digram

The solutions found at the four points of $Q_1 \times Q_2$ are used to determine the solutions at their neighbors in $Q_1 \times Q_2$. The resulting solutions are then used in a similar manner. This is continued until the solutions have been established for all points of $Q_1 \times Q_2$ or until failure to converge exhausts the untried neighbors. Given the initial solutions, this technique finds the solutions at the other points of $Q_1 \times Q_2$ in an efficient manner.

3. For the points of $Q_1 \times Q_2$ for which solutions are found, the remaining characteristics are computed and saved. This results in $p_3, p_4, \dots, p_n, c_{n-1}, c_n, c_{n+1}, \dots, c_m$ for each solution being expressed as a function of p_1 and p_2 defined over $Q_1 \times Q_2$.
4. An interface is provided for programs doing additional analysis and/or computations using the arrays of system characteristics as input.
5. A contour plotting capability is provided to plot the functions of two variables described in 3. It uses interpolation to approximate these functions over $D_1 \times D_2$. The methods used allow for the possibility that a particular solution is not determined for every point of $Q_1 \times Q_2$. In this case contours can be plotted over whatever subregions the available points of $Q_1 \times Q_2$ permit.

Methods

The methods used to perform the functions described above are as follows:

1. For convenience define $f_i = c_i - k_i, i = 1, 2, \dots, n-2$ and let $F = (f_1, f_2, \dots, f_{n-2})$; then $F=0$ means $c_i = k_i, i = 1, 2, \dots, n-2$. Let $U = \sum_{i=1}^{n-2} f_i^2$ and note that $U = 0$ if and only if $F=0$. In order to find solutions, the Secant Method [1] for solving systems on nonlinear

functions is used in conjunction with a search strategy* designed to minimize the number of system evaluations required. The strategy is as follows:

- a. Compute and save the value of U for each point of $Q_3 \times Q_4 \times \dots \times Q_n$.
- b. Find the smallest U not yet tried and use the corresponding point as the initial point for the Secant Method.
- c. If the method converges to a new solution, save it; if the method moves toward a solution previously found, stop iterating; if the method fails to converge, record the number of successive failures.
- d. Repeat steps b and c until the number of successive failures is greater than 10% of the number of points as yet untried.

A second search method (Total Domain Search Method) is available which starts the Secant Method at every point of $Q_3 \times Q_4 \times \dots \times Q_n$. This method is a forerunner of the above method and is considerably less efficient in terms of system evaluations. It remains in the program in case all else fails or as a basis for comparison. Provisions are made for other search and iterative techniques that may be supplied by future users for their particular applications.

2. Solution mapping [2] is used to map each initial solution found to as many points of $Q_1 \times Q_2$ as possible. This is a technique whereby the extrapolation of the solutions found at two adjacent points is used as an initial guess for an iterative procedure in an effort to converge to solutions for the points on either side of the adjacent set. In the absence of adjacent sets, solutions at the four neighboring points are used as initial guesses.

*Referred to hereafter as "Optimum Domain Search" method.

3. For each grid* point (p_1, p_2) and the corresponding solutions (p_3, p_4, \dots, p_n) , the system characteristics are computed and stored.
4. To perform functions special to a particular application of PAP, control is transferred to the user supplied subroutine AUXPRO.
5. The set of grid points* for which a solution is found and the corresponding non-grid parameter and characteristics represent a number of functions of two variables. Two-directional parabolic interpolation is used in a scheme using nine grid points at a time to find the contours of the specified functions. If less than nine points are available, various combinations of parabolic and linear interpolation are used as the number of available grid points permits. Subroutine INGRID [4] generates the contours and PLTCON [3] plots them.

A number of features of the program that make it flexible and easy to use are:

1. The program structure provides for additional search methods and different methods for solving systems of nonlinear equations. Also, given the appropriate search and iterative methods, a more general problem may be studied using the program: Let H be a continuous function of P . For a given point of $D_1 \times D_2$, find the points of $D_3 \times D_4 \times \dots \times D_n$ such that $F(P) = 0$ and $H(P)$ is optimum. This problem requires that $0 \leq r < n - 2$ where r is the number of equality constraints ($r=0$ means no equality constraints are imposed).
2. It is possible to stop the program after any of the five major functions have been performed, saving the data necessary to restart the program at this point.

*The "set of grid points" or "the grid" means the set $Q_1 \times Q_2$.

3. A number of different grids and the corresponding characteristics can be made available to AUXPRO simultaneously.*
4. It is possible for the user to specify most of the input variable names. This makes it possible for him to select a set of input variables that are natural to his application and it could reduce the number of inputs required.*

The Simulator

The user must supply a number of procedures (PROC's) and subroutines which simulate the system under study and perform various I/O functions. When the procedures have been written as specified below, the main program, PAP, and subroutine RPD, must be compiled with the PDP element containing these PROC's. This creates relocatables for PAP and RPD which are particular to the application at hand. These relocatables, the relocatables for the other PAP subroutines, and the relocatables for the user's subroutines can then be collected into an absolute element and executed. The size of the program requires overlays, the modularity of the program makes this easy to do. This will be discussed in the Program Description section.

Description of User Supplied PROC's

The user may not need to make use of all the PROC's defined below; however, as PAP references each PROC name, at least a blank PROC by that name must be provided.

All but one of the Users PROC's are included (by a FORTRAN V Include statement)** in the main program PAP. These are discussed in the order they are included.

The first two PROC's contain documentation only. They are included at appropriate places to make the introductory commentary complete. Their suggested use is as follows:

RUNDOC	contains the names and definitions of the input variables in namelist RUNDAT which is specified by the user in PROC RUNNL defined below.
--------	--

*See [6] for an example of a use of this feature.

**See "INCLUDE Statements", UNIVAC 1108 FORTRAN V Programmers Reference Manual.

SAVDOC contains the names and definitions of the input variables in namelist SAVDAT which is specified by the user in PROC SAVNL defined below.

The next procedure provides the means of communication between PAP and the user's subroutines.

KOMUSE contains labeled common USE or other labeled commons not used by PAP and its subroutines.

The next two PROC's specify two namelists which are to be used as the means of defining all of the control variables required by the simulator and PAP except for the plot control variables which are read by namelist PLTDAT specified in PAP. The variables required by PAP are defined in Tables 1 and 2. Variables KON, IUNIT, ICHSA, and NFR must appear explicitly in namelist RUNDAT. The other variables in Tables 1 and 2 may appear in either namelist RUNDAT or SAVDAT or they may be defined in SIMDFN (a PROC defined below) using variables read by these two namelists. The PROC's specifying namelists RUNDAT and SAVDAT are defined as follows:

RUNNL contains namelist RUNDAT which is used to read those input variables relevant to the current run only.

SAVNL contains namelist SAVDAT which is used to read all input variables that may be needed in case of restart. These include the variables that control the simulation, analysis, and define the constraints. This namelist is saved each time a data set is saved.

The next PROC is included before any executable statements. It may be used to initialize variables at the beginning of a run, or to read in a data base. It may also be used to set the values of various inputs that are not often changed. These inputs can be changed when necessary by the subsequent reading of namelist RUNDAT, SAVDAT, and PLTDAT. In summary,

INITIL contains the FORTRAN code that initializes various variables at the beginning of a run.

The next PROC is included after namelist RUNDAT, SAVDAT, and PLTDAT have been read. It may be used to derive from the variables read by namelists RUNDAT and SAVDAT the specific variables required by the simulation and PAP. If all of the control variables are defined in the namelists in

just the form required by PAP and the simulation, this procedure can be left empty. However, it is generally useful to choose the input variables in a way that make their functions easy to remember and the specification of a data case as simple as possible, letting the computer translate them to the exact form required by the program.

SIMDFN contains the FORTRAN code which performs this translation.

The last PROC is included in subroutine RPD which reads namelist SAVDAT from the logical unit specified in its argument list. None of the information read is passed to other parts of the program. The need for this subroutine arises when plot data is to be read from more than one logical unit. Namelist SAVDAT is at the beginning of all these data files but only one reading of the namelist is required by the program. Subroutine RPD is called to read pass namelist SAVDAT. The PROC required by RPD is defined as follows:

RPDDS contains a dimension statement for all variables in namelist SAVDAT. The dimension and size of each array must correspond to its dimension and size in PAP. This dimension statement may use the parameter statement found in the PROC PARSTA (see Appendix C for PARSTA listing).

Description of User Supplied Subroutines

Let CS be the vector of differences between the constraint functions and their desired (target) values.* A subroutine must be provided which computes CS as a function of the given values of the parameters, P. The subroutine's entry point must be

CONSTR(P,CS).

A subroutine must be provided which computes the characteristics, C, of the system for given values of the parameters, P. This subroutine's entry point must be

CHARAC(P,C).

*CS corresponds to $F = (f_1, f_2, \dots, f_{n-2})$ discussed earlier.

The next subroutine whose entry point is

PRTCHA(P,C)

must be provided to print P and C in a format that suits the user.

The last subroutine whose entry point is

AUXPRO

may be used as an interface for other subroutines that perform functions that use the data generated by PAP and/or the plotting capability it provides.

Appendix B provides listings for the procedures and subroutines required by the test case discussed later.

Usage

The five major functions of PAP are performed respectively, by the five subroutines: INISOL, FILGRD, COMCHA, AUXPRO, and PLTCON. Each of these subroutines, except INISOL, requires data generated by some one of the other subroutines. The data generated by a call to one of the five major subroutines is called a data set for that subroutine. In order to start the program at any subroutine, other than INISOL, the necessary data set must have been generated and saved. Data sets for up to three calls each to INISOL, FILGRD, and COMCHA and two calls to AUXPRO can be saved during any execution of PAP. These data sets can subsequently be made available to the appropriate subroutines for further processing later in the same execution or in subsequent executions. The data sets are saved on logical units as follows:

<u>Subroutine</u>	<u>Logical Units</u>
INISOL	8, 9, 10
FILGRD	11, 12, 13
COMCHA	14, 15, 16
AUXPRO	17, 18

Figure 1 is a functional flowchart of PAP showing all of the possible paths through the program. Taking one of these paths is said to be a pass. A pass during which the data set from a particular major subroutine is saved is called

a save pass for that subroutine. A single pass may be a save pass for more than one major subroutine. Any number of passes may be made during any execution as long as the number of save passes does not exceed the limits stated earlier, i.e., three save passes each for INISOL, FILGRD, COMCHA, and two save passes for AUXPRO.

The functional blocks of Figure 1 are numbered 1 through 5. The return to the beginning of the program is numbered 6 and EXIT is numbered 7. These numbers are used to define the sequence of subroutine calls during the course of an execution. The user determines the sequence by setting an input variable (KON).

Figure 2 shows how data cases are read. If a pass begins at INISOL, namelists RUNDAT and SAVDAT are read from cards in that order. If a pass begins at any other point, RUNDAT is read from cards, SAVDAT is then read from IUNIT (an input variable), and finally at the option of the user, SAVDAT may be reread from cards. This feature gives the user the ability to stop a pass after any subroutine, save the necessary data set (namelist SAVDAT will be saved also), and studying the results before deciding whether or not to continue. If it is decided to continue, the data in namelist SAVDAT may be used as it was stored, or it may be altered by reading SAVDAT from cards before continuing. Namelist PLTDAT is read once for each frame to be generated.

Definition of PAP Input Variables

The names and descriptions of all input variables required by PAP are given in Tables 1, 2, and 3. Tables 1 and 2 contain all the PAP input variables that must be defined by means of namelists RUNDAT and SAVDAT. These two namelists must be specified by the user in PROC's, RUNNL and SAVNL. The variables KON, IUNIT, ICHSA, and NFR must appear explicitly in namelist RUNDAT. The remaining variables described in Tables 1 and 2 may appear in either namelist explicitly or, they may be defined by user supplied FORTRAN code in PROC's INITIAL or SIMDFN. If some of the inputs to PAP do not change for a particular application, then they may be set once and for all in PROC INITIL. Those variables that change from run to run and therefore depend on the input variables read by namelists RUNDAT and SAVDAT may be defined by user supplied FORTRAN code in the PROC SIMDFN. Input variables required to define and control the user's simulation may be read using these namelists also.

Variables defined by means of namelist RUNDAT should pertain to the bookkeeping details of the run. Table 1 gives the PAP input variables usually defined by means of this namelist. Namelist SAVDAT is saved each time a data set is saved. It should contain the variables which define the type of simulation being done, the constraints being satisfied, and the number and bounds of the parameters. It should also contain any other variables that may be needed to identify the data or to restart the program later.

Namelist PLTDAT is specified in PAP and contains the variables required to define a single frame that is to be plotted. It is read NFR times in sequence storing the information so that it is related to the particular frame being defined. None of the input variables are distributed, therefore, a subsequent namelist PLTDAT need only contain the variables that differ from the preceding one.

TABLE 1

<u>Name</u>	<u>Type/Size</u>	<u>Description</u>
KON	INT/8	Control sequence indicator. KON(J) = I ($1 \leq I \leq 5$) causes subroutine I* to be called Jth in the sequence. KON(J) = 6* returns control to the beginning of the program. KON(J) > 7* causes normal termination of the execution.
IUNIT	INT/	Initial data unit. Defines the logical unit from which SAVDAT and the appropriate data set are read if the program is to be stated at a point other than INISOL, i.e., if KON(1) > 1.
ICHSA	INT/	Data change indicator. If ICHSA = 1 and KON(1) > 1, namelist SAVDAT is read from logical unit 5 after it is read from logical unit IUNIT. If ICHSA \neq 1 and KON(1) > 1, SAVDAT is read from IUNIT only. When KON(1) = 1, ICHSA has no effect.

*See Figure 1.

TABLE 1 (cond.)

<u>Name</u>	<u>Type/Size</u>	<u>Description</u>
NFR	INT/	Number of frames to be plotted. Causes namelist PLTDAT to be read NFR times.
NF14 NF15 NF16 NF17 NF18	INT/	Plot data file flags. Plot data is to be read from the logical unit indicated by the numerical suffix of the variable name when that variable is set to 1. If only one logical unit is being used, these variables may be ignored.
ISAI ISAF ISAC ISAA	INT/	Data set save flags. These flags when set to 1 cause the data set of the subroutine whose initial is the same as the last letter of the variable name to be saved.
IPRI	INT/	INISOL print flag. The solutions found are always printed. If IPRI = 1, the search progress is also printed.
IPRF	INT/	FILGRD print flag. A display of the grid points for which solution were found and the percentage of points converging is always printed. If IPRF = 1, solution mapping progress is printed also.
IPRC	INT/	COMCHA print flag. = 0, no results printed. = K, ($K \geq 1$) subrouting PRTCHA is called at the corner points of the grid and at points of the grid defined by the intersections of the Kth, 2Kth, 3Kth, ..., grid lines.
IPRA	INT/	AUXPRO print flag. For use by the user.

TABLE 1 (cond.)

<u>Name</u>	<u>Type/Size</u>	<u>Description</u>
IPRP	INT/	PLTCON print flag. ≤ 0 , no printing. ≥ 1 , frame summary table, drum storage information, and function arrays printed. ≥ 2 , function value and availability arrays printed by INGRID. ≥ 3 , point selection map printed by INGRID.

TABLE 2

<u>Name</u>	<u>Type/Size</u>	<u>Description</u>
NP	INT/	Number of parameters (independent variables).
A	REA/10	Lower bounds for the domains of the parameters.
B	REA/10	Upper bounds for the domains of the parameters.
DELP	REA/10	Step size for the parameters. DELP(1) and DELP(2) determine the grid mesh. DELP(3), DELP(4), ..., DELP(NP) determine the search step size.
PlI	REA/	The value of P(1) while the parameter space of P(3), P(4), ..., P(NP) is being searched. $A(1) \leq PlI \leq B(1)$.
P2I	REA/	The value of P(2) while the parameter space of P(3), P(4), ..., P(NP) is being searched. $A(2) \leq P2I \leq B(2)$.
NCS	INT/	The number of equality constraints to be satisfied, $NCS = NP - 2^*$.
EPS	REA/8	The maximum acceptable differences between the constraint functions and their target functions.
ISM	INT/	To use the Optimum Domain Search, set ISM=3. To use the Total Domain Search, set ISM=1. Provisions are made for three other techniques that may be required by future users. For these, use ISM=2, 4, or 5.

*At present this variable is superfluous. However, if optimization subject to equality constraints is implemented, it can be used as follows: $0 \leq NCS \leq NP - 2$, where $NCS=0$ implies optimization only; $NCS=NP-2$ implies equality constraints only; and, $0 < NCS < NP-2$ implies optimization subject to NCS equality constraints.

TABLE 2 (cond.)

<u>Name</u>	<u>Type/Size</u>	<u>Description</u>
ICM	INT/	Iterative procedure selector. = 3, the secant method is used (subroutine CONV3). Provisions are made for four other methods which may be required by future applications. For these, use ICM = 1, 2, 4, or 5.
NSG	INT/	Number of initial solutions to be mapped. $NSG \leq 5$.
ISGR	INT/5	If the Ith grid is to be mapped using the Jth solution as a starting point, set ISGR(I) = J. This grid will be identified by the number of the solution from which it is mapped.
NCG	INT/	The number of grids for which characteristics are to be computed.
ICCH	INT/5	If the Ith set of characteristics to be computed uses the grid mapped from the Jth solution, set ICCH(I) = J. Note that the Jth solution must have been mapped.
NC		Number of characteristics computed for each grid point.

TABLE 3

<u>Name</u>	<u>Type/Size</u>	<u>Description</u>
IOS	INT/	Plotting option selector. = 0, off-line plots only. = 1, printer plots only. = 2, both types.
FTLT	HOL/8	48 character frame title.
ALAB	HOL/8	48 character abscissa label.
OLAB	HOL/8	48 character ordinate label.
LABC	INT/	Not used at present.
AMI	REA/	Minimum value of the frame abscissa.
AMA	REA/	Maximum value of the frame abscissa.
OMI	REA/	Minimum value of the frame ordinate.
OMA	REA/	Maximum value of the frame ordinate.
NFU	INT/	Number of functions on this frame (≤ 10).
IUN*	INT/10	Logical unit indicators. IUN(J) = I means that the Jth function of this frame is to be read from logical unit I.
ISO	INT/10	Solution numbers. ISO(J) = I means that the Jth function of this frame is from the Ith solution.

*The remaining variables in this table require a value corresponding to each function to be plotted on the present frame.

TABLE 3 (cond.)

<u>Name</u>	<u>Type/Size</u>	<u>Description</u>
IVA	INT/10	Variable numbers. IVA(J) = I means that the Jth function of this frame is the Ith variable of the solution.
PlIN	REA/10	Interpolation increments for the first independent variables. DELP(1)/PlIN should be a positive integer.
P2IN	REA/10	Interpolation increments for the second independent variables. DELP(2)/P2IN should be a positive integer.
ICD	INT/10	Contour plane indicators. = 0, constant function contours. = 1, first variable constant contours. = 2, second variable constant contours.
IAOS	INT/10	Abcissa-ordinate selectors. = 0, P(1)/P(2), P(2)/F, or P(1)/F is the abscissa/ordinate. = 1, the reverse is true.
CMI	REA/10	Minimum contour values (used when ICD=0).
CMA	REA/10	Maximum contour values (used when ICD=0).
CIN	REA/10	Step size between contours (used when ICD=0).
INS	INT/10	Indices of the initial contours (used when ICD=1 or 2).
LAS	INT/10	Indices of the final contours (used when ICD=1 or 2).
ISI	INT/10	Increments of indices (used when ICD=1, or 2).

Test Case

It should be pointed out and emphasized that PAP does not require that the system under study be represented by a set of closed form equations. On the contrary, the most valuable applications of PAP are those which require considerable effort to compute the system characteristics and constraints, e.g., see Reference [6]. However, to test and to illustrate the use of PAP, a contrived system which is represented by a set of closed form equations was chosen. This avoids the needless confusion introduced by a complicated simulation and allows full attention to be given to the method.

To simplify notation let $(u,v,w,x,y,z) = (p_1, p_2, \dots, p_6)$. Consider the system described by the following characteristics:

$$c_1 = wx e^{(yz+uv)}$$

$$c_2 = 10wx \sin(yz+uv)$$

$$c_3 = wy^2 + 5xz + uv$$

$$c_4 = 10wxyz - uv$$

$$c_5 = u^2 + v^2 + (1-w)^2 + (1-x)^2 + y^2 + z^2$$

Let

$$f_1 = c_2 - 9.093$$

$$f_2 = c_3 - 7.0$$

$$f_3 = c_4 - 9.0$$

$$f_4 = c_5 - 4.0$$

Let the interval domains of the parameters be $[0,1.4]$, $[0,1.4]$, $[.75, 1.25]$, $[.75, 1.25]$, $[.75, 1.25]$, and $[.75, 1.25]$; and let $\Delta p_i = (.2, .2, .25, .25, .25, .25)$.

With $p_1 = 1.$ and $p_2 = .75$, $D_3 \times D_4 \times \dots \times D_6$ was searched. Two solutions were found and mapped to as many points of $Q_1 \times Q_2$ as possible. Appendix A contains selected pages of the output and a number of the resulting plots. Appendix B contains listings of the test case simulation subroutines and PROC's.

Program Description

Listings for the following are given in Appendix C:

PAP	COMCHA	VCP
PAPSPC	PLTCON	CONV3
RPD	SEEK1	GAUSS
INISOL	SEEK3	INPR
FILGRD	CONV	GRIDMK
		FRMPLT

The introductory commentary contained in these subroutines describes their functions (see Purpose) as related to the overall program. Subroutines PLTCON and FRMPLT are slight variations of those documented by [3, 4, 5]. The subroutines they reference are unchanged and are described in the above references. Two other subroutines are referenced - OCLOCK and RKPLOT; they are available from the Bellcomm Applications Program Library.

When executing the test case, the following overlay scheme required a maximum of $(41000)_{10}$ words of core storage:

PAP

INISOL

FILGRD

COMCHA

AUXPRO

PLTCON


P. F. Long

1032-PFL-prs

Attachments

BELLCOMM, INC.

REFERENCES

1. "The Secant Method for Simultaneous Nonlinear Equations", Philip Wolfe, Comm. A.C.M. 2 (12), 1956.
2. "Solution Mapping - a Method of Parametric Analysis", P. F. Long, Bellcomm Memorandum for File, April 16, 1968.
3. "Description of PLTCON - Contour Plotting Program", P. F. Long, Bellcomm Memorandum for File, June 25, 1969.
4. "INGRID - A Contour Generating Program", M. P. Odle, Bellcomm Memorandum for File, June 30, 1969.
5. "Computer Program Abstract and Write-Up for SCPLOT - A General Purpose Interface to the S-C 4020 Plotter", O. R. Pardo, Bellcomm Memorandum for File, August 18, 1969.
6. "INTAP - Interplanetary Trajectory Analysis Program", R. W. Grutzner, Bellcomm Technical Memorandum (in preparation).

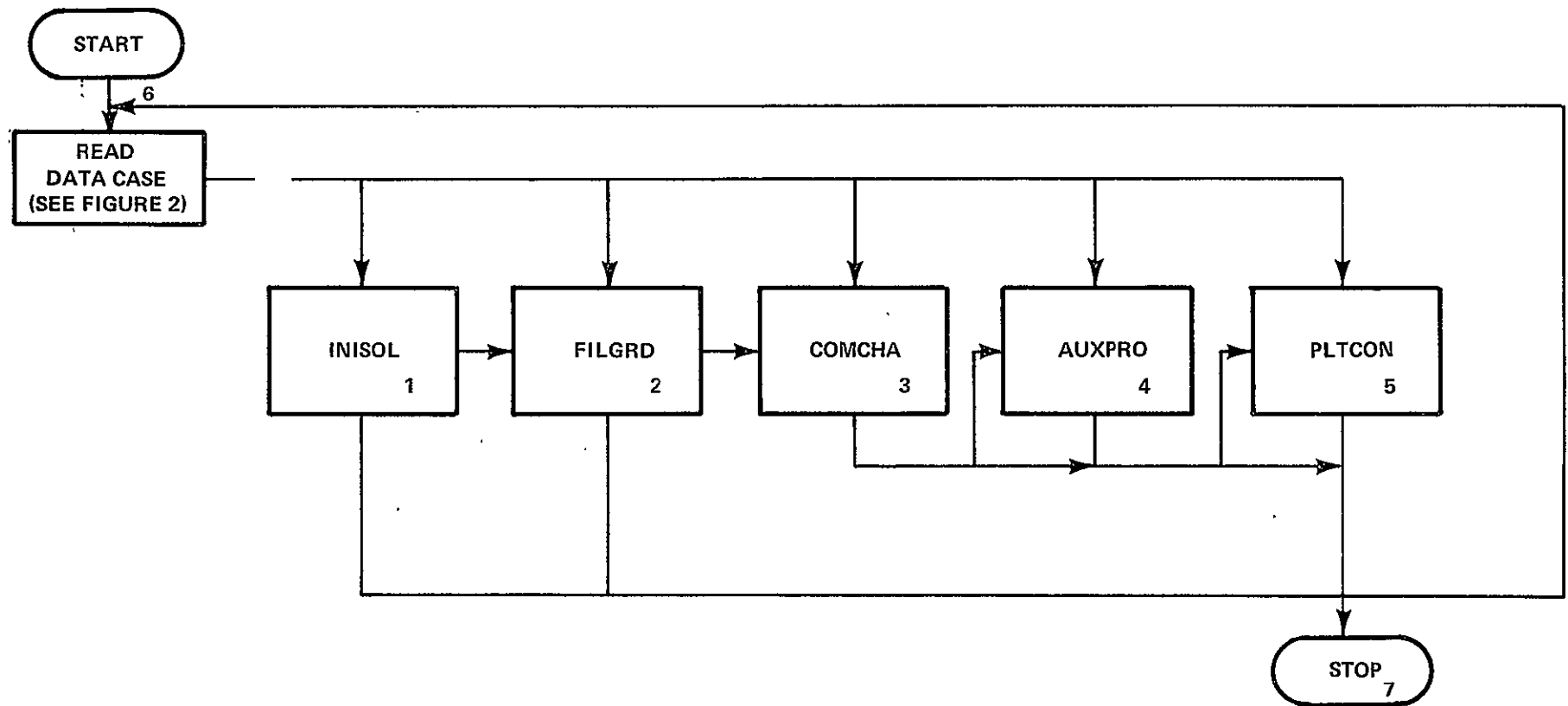


FIGURE 1 - FUNCTIONAL FLOWCHART OF PAP

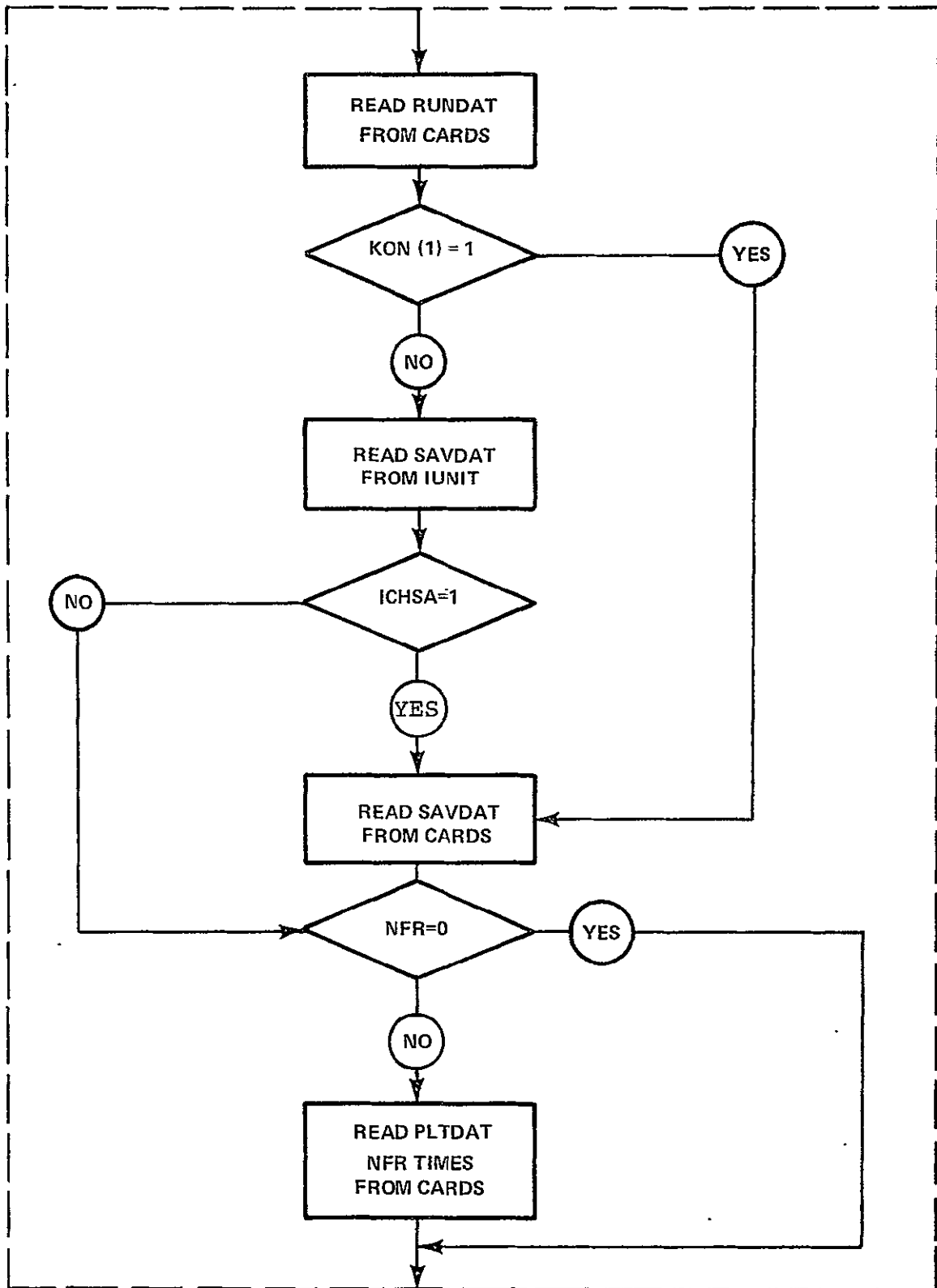


FIGURE 2 - READ DATA CASE

APPENDIX A

Selected Pages of
Test Case Output and Plots

STANDARD 4-L PAP TEST

DATE 122369

PAGE

3

```

M      AOT      BCM11,PAPLNK
K01=1,2,3,5,7,IFN=5,
IPAP=1,
IS1=3,IC1=3,
$E1L
IS0=2,IC0=2,ISGR=1,2,ICCH=1,2,
A=2*.0,4*.75,
U=2*1.4,4*1.25,
DEL P=2*.2,4*.25,
P11=1.,P1=1.75,
MP=6,ICS=4,IC=5,
EPS=4*.0001,
IFS=2,4,5,3,
TV=9.093,9.,4.,7.,
$END
FTLT=' P(3) OF SOLUTION 1 ',
IUS=2,ALAB=' P(1) ',OLAP=' P(2)
A4A=1.4,ONA=1.4,NFU=1,IUN=26,ISO=1,IVA=1,
P11N=.01,P21N=.01,CM1=.5,CMA=.9,CIN=.05,
$END
FTLT=' P(4) OF SOLUTION 1 ',
IVA=2,CIN=1.,CMA=2.5,CIN=.2,
$END
FTLT=' P(5) OF SOLUTION 1 ',
IVA=3,CM1=.9,CMA=1.6,CIN=.1,
$END
FTLT=' P(6) OF SOLUTION 1 ',
IVA=4,CM1=.4,CMA=1.1,CIN=.1,
$END
FTLT=' C(1) OF SOLUTION 1 ',
IVA=5,CM1=2.,CMA=4.,CIN=1.,
$END

```

*****CALL INISOL AT TIME 13:24:43.545*****

2 SOLUTION(S) FOUND

1	.10000000+01	.75000000+00	.75656846+00	.12328266+01	.12920982+01	.80901879+00
2	.10000000+01	.75000000+00	.11190744+01	.83346562+00	.78293543+00	.13351467+01

*****SYSTEM EVALUATED 135 TIMES IN INISOL*****

*****CALL FILGRD AT TIME 13:25:27.253*****

*****MAP SOLUTION NO. 1*****

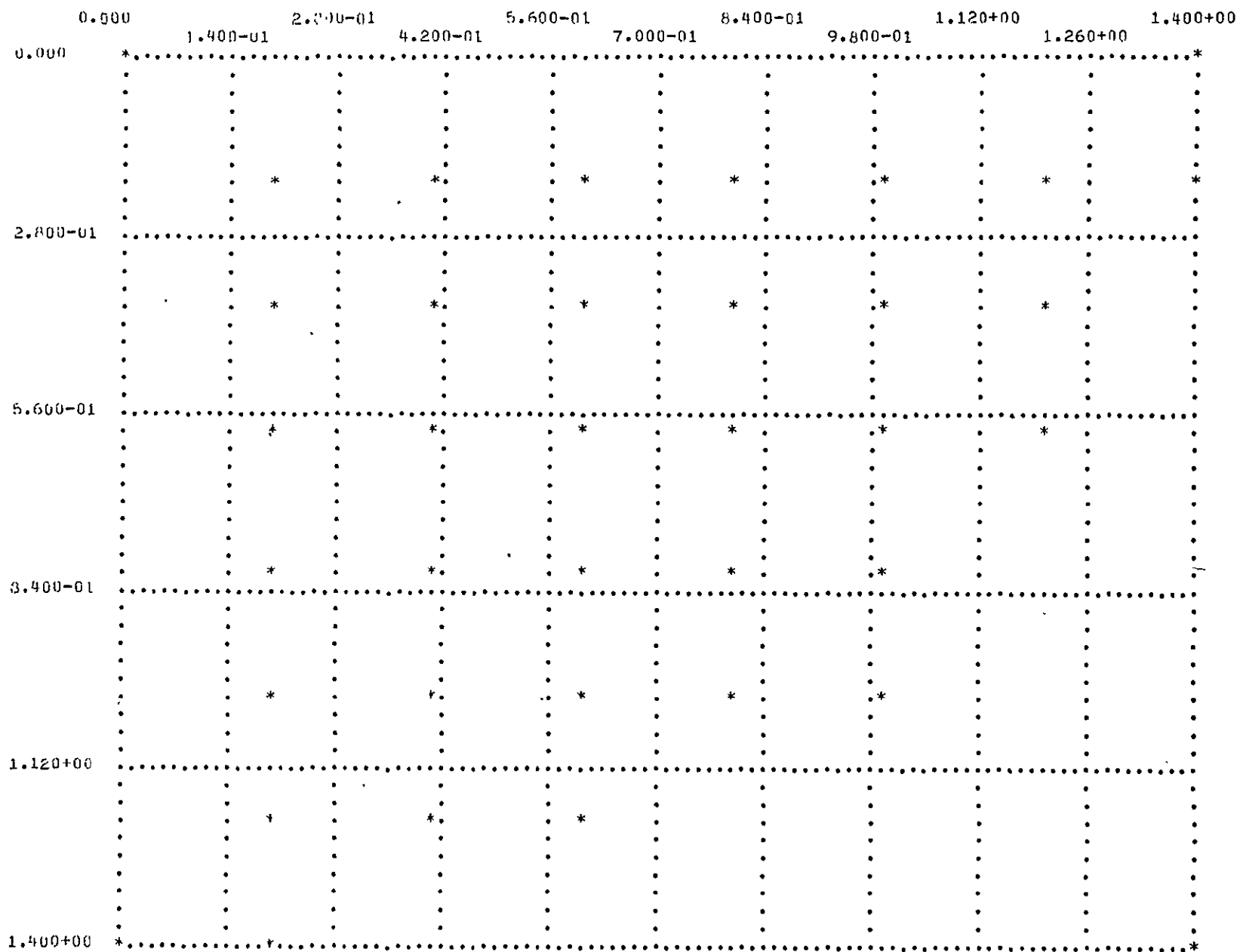
*****PERCENTAGE OF POINTS CONVERGING = 51.563*****

*****PERCENTAGE OF POINTS CONVERGING = 51.563*****

STANDARD 4-D PAI' TEST

DATE 122369

PAGE 4



POINTS FOR WHICH SOLUTIONS WERE FOUND - P1 VERTICAL AXIS, P2 HORIZONTAL AXIS, PERCENTAGE 51.563

STANDARD 4-L PAP TEST

DATE 122369

PAGE 8

FILE NO.	PLOT OP.	AMIN	AMAX	OMIN	OMAX								
		UNIT	SOL.NO.	VAR.NO.	XY OP.	DELP1	DELP2	CMIN	CMAX	CDEL	IVS	IS	LS INC
1	2	.0	1.4	.0	1.4								
		26	1	1	0	.01	.01	.50	.90	.05	0	0	0 0
2	2	.0	1.4	.0	1.4								
		26	1	2	0	.01	.01	1.00	2.50	.20	0	0	0 0
3	2	.0	1.4	.0	1.4								
		26	1	3	0	.01	.01	.90	1.60	.10	0	0	0 0
4	2	.0	1.4	.0	1.4								
		26	1	4	0	.01	.01	.40	1.10	.10	0	0	0 0
5	2	.0	1.4	.0	1.4								
		26	1	5	0	.01	.01	2.00	8.00	1.00	0	0	0 0

STANDARD 4-D PAP TEST.

DATE 122369

PAGE 10

FUNCTION ARRAY 1 FROM UNIT 26 SOLUTION 1 AND VARIABLE 1

NP1 = 8 NP2 = 8

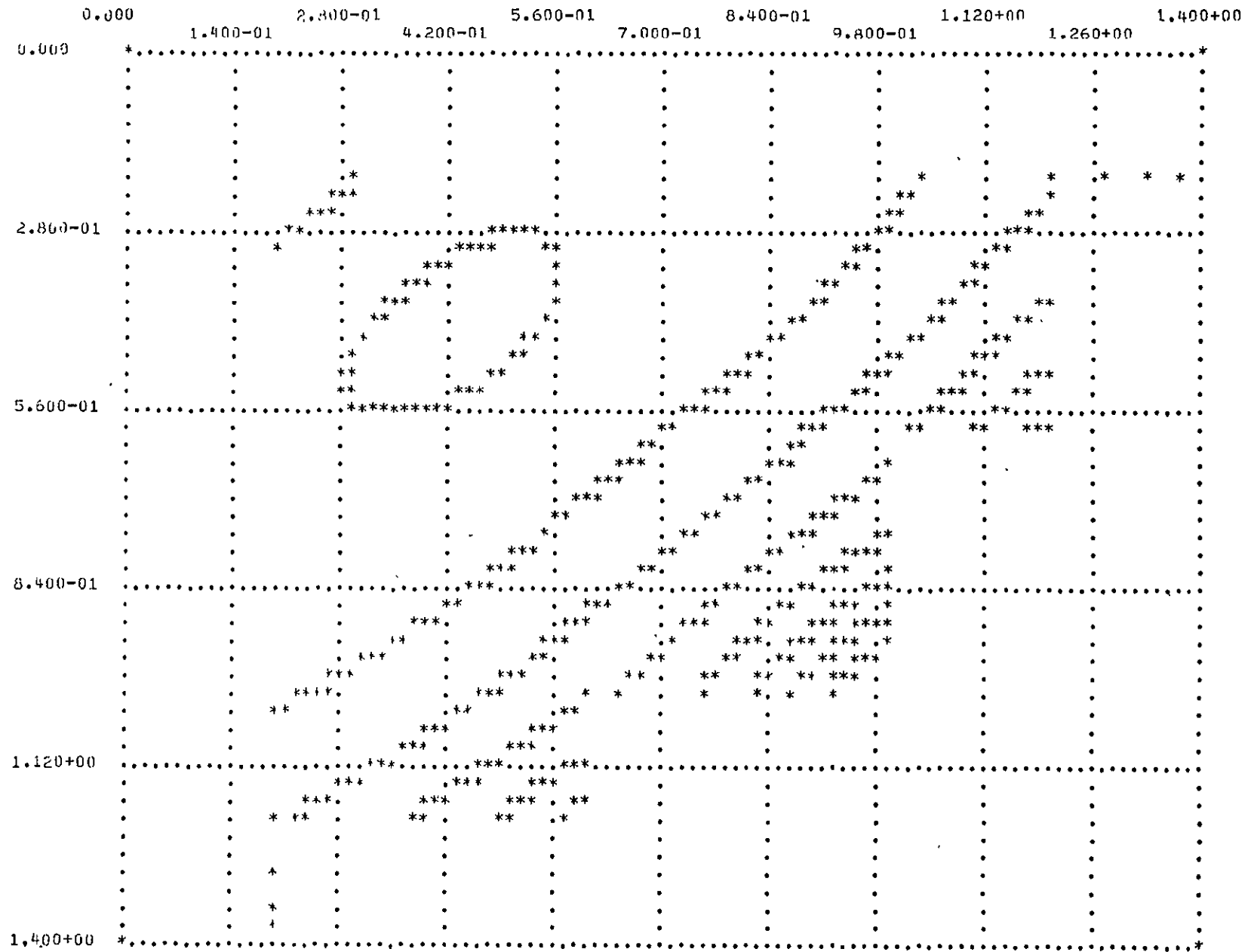
P1(.) =	.00000000	.20000002+00	.40000000+00	.60000000+00	.80000000+00	.10000000+01	.12000000+01	.14000000+01
P2(.) =	.00000000	.20000002+00	.40000000+00	.60000000+00	.80000000+00	.10000000+01	.12000000+01	.14000000+01
F(1,.) =	.00000000	.00000000	.00000000	.00000000	.00000000	.00000000	.00000000	.00000000
F(2,.) =	.00000000	.64045120+00	.57353177+00	.56043664+00	.56872863+00	.59407602+00	.64463036+00	.82982851+00
F(3,.) =	.00000000	.57353178+00	.54587185+00	.55534592+00	.58195781+00	.62631900+00	.70664830+00	.00000000
F(4,.) =	.00000000	.56048349+00	.55534592+00	.58141367+00	.62483601+00	.68996587+00	.81553596+00	.00000000
F(5,.) =	.00000000	.56872645+00	.58195775+00	.62483587+00	.68842107+00	.78383573+00	.00000000	.00000000
F(6,.) =	.00000000	.59407719+00	.62632402+00	.68996467+00	.78385219+00	.99851991+00	.00000000	.00000000
F(7,.) =	.00000000	.64465158+00	.70664390+00	.81554517+00	.00000000	.00000000	.00000000	.00000000
F(8,.) =	.00000000	.82981560+00	.00000000	.00000000	.00000000	.00000000	.00000000	.00000000

*****CALL TO INGRID FOR INTERPOLATION*****

STANDARD 4-D PAP TEST

DATE 122369

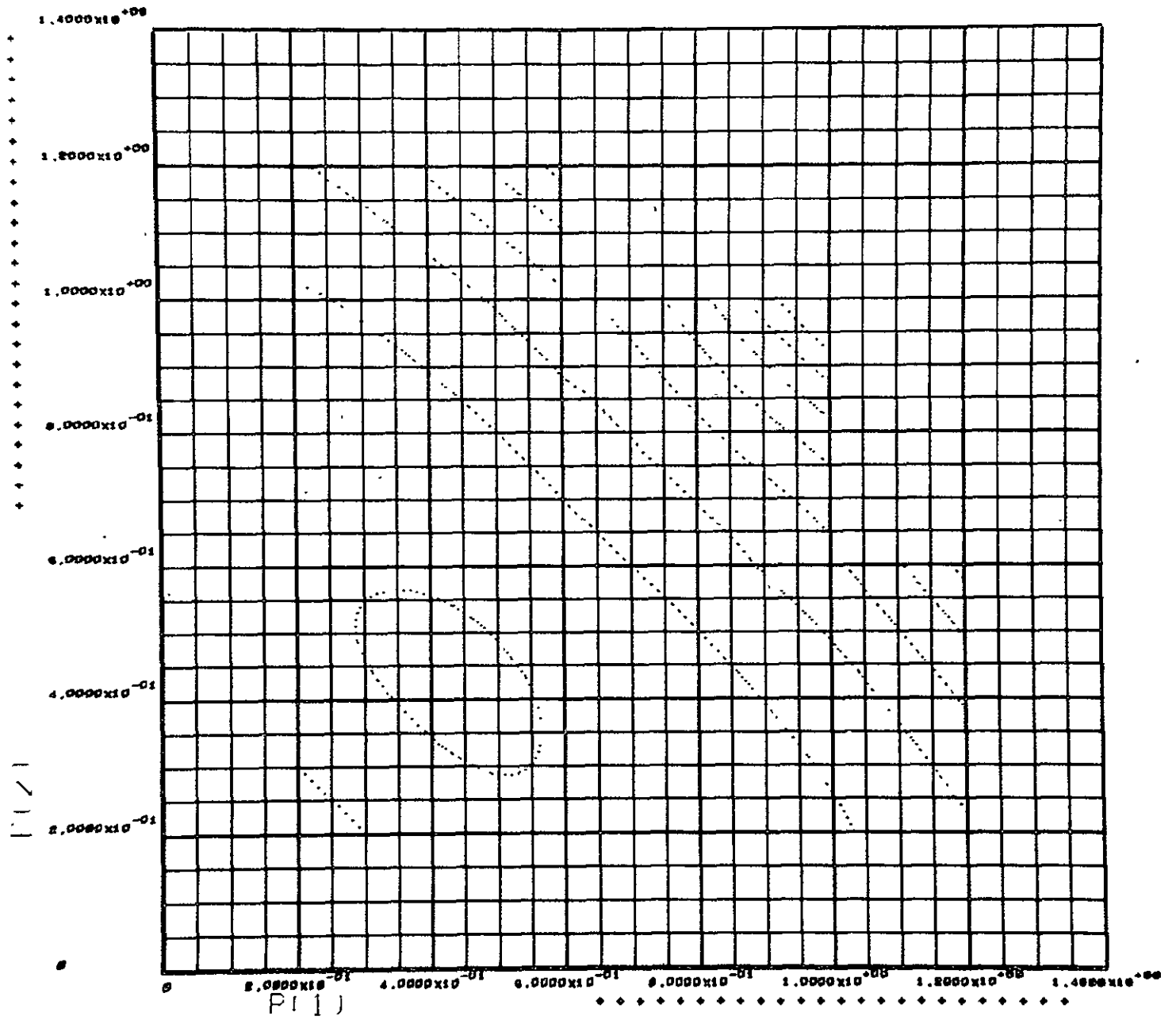
PAGE 11



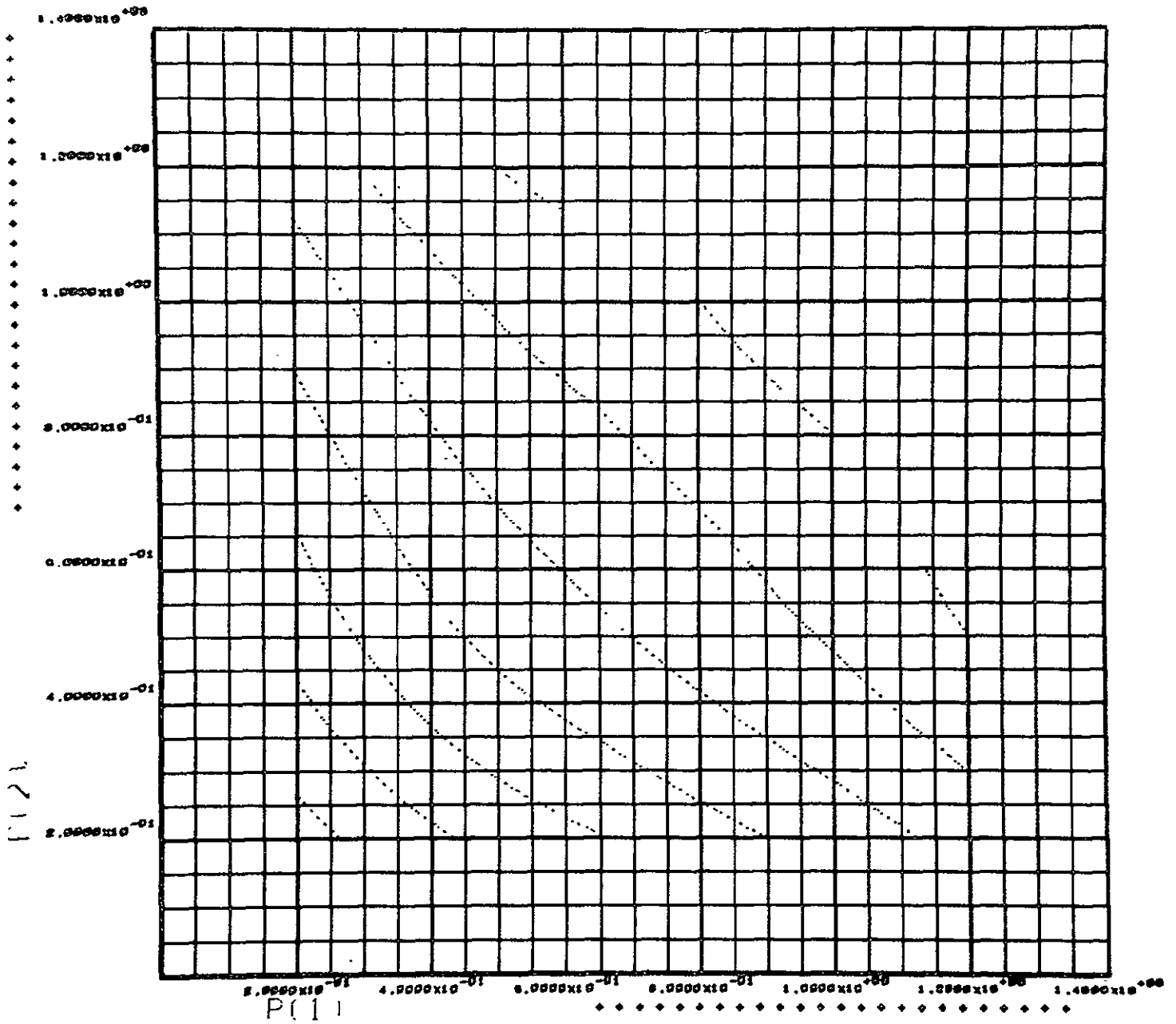
RUN OF 144

PL. 11 IN 1

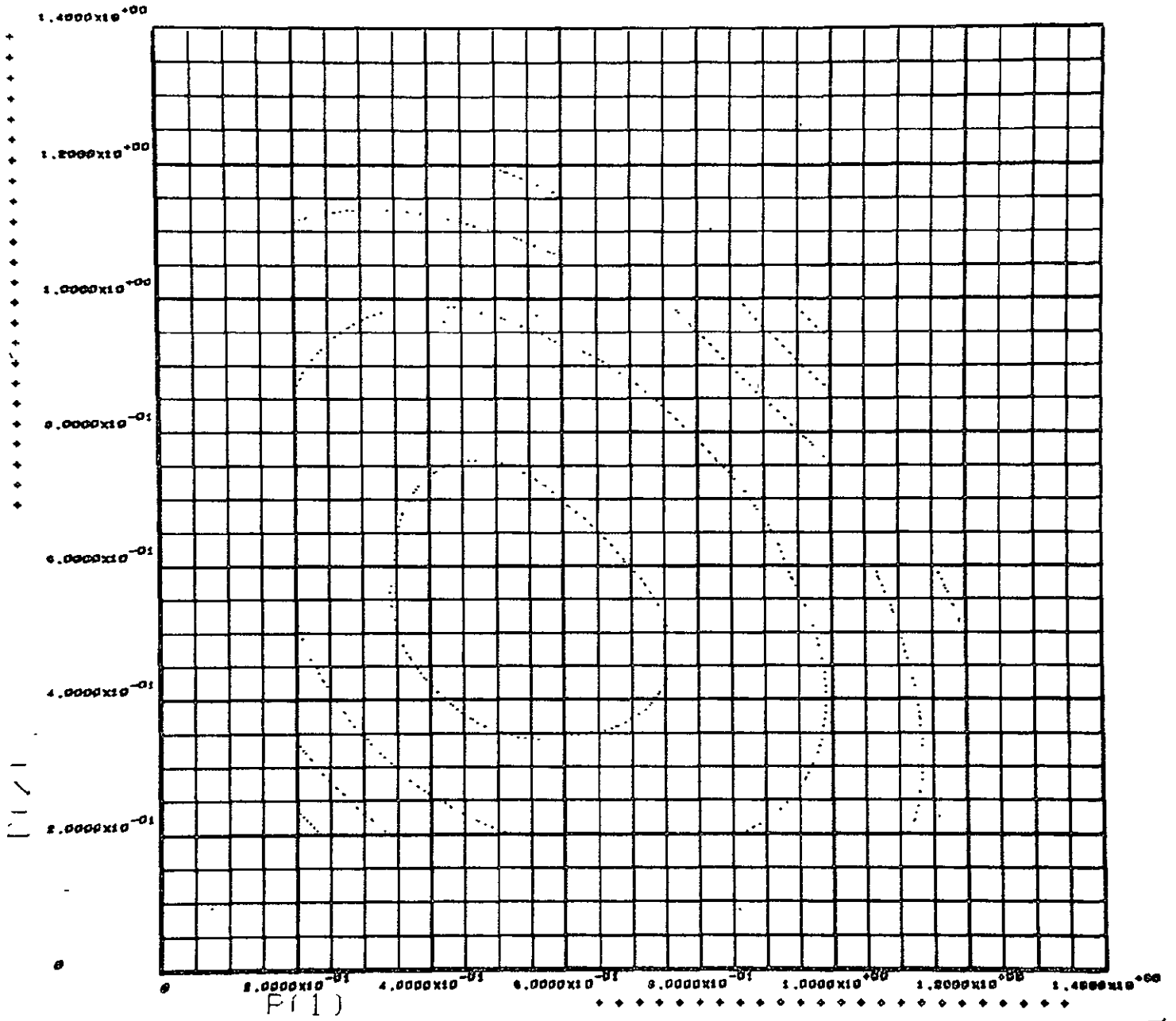
*****123069*****123069*****



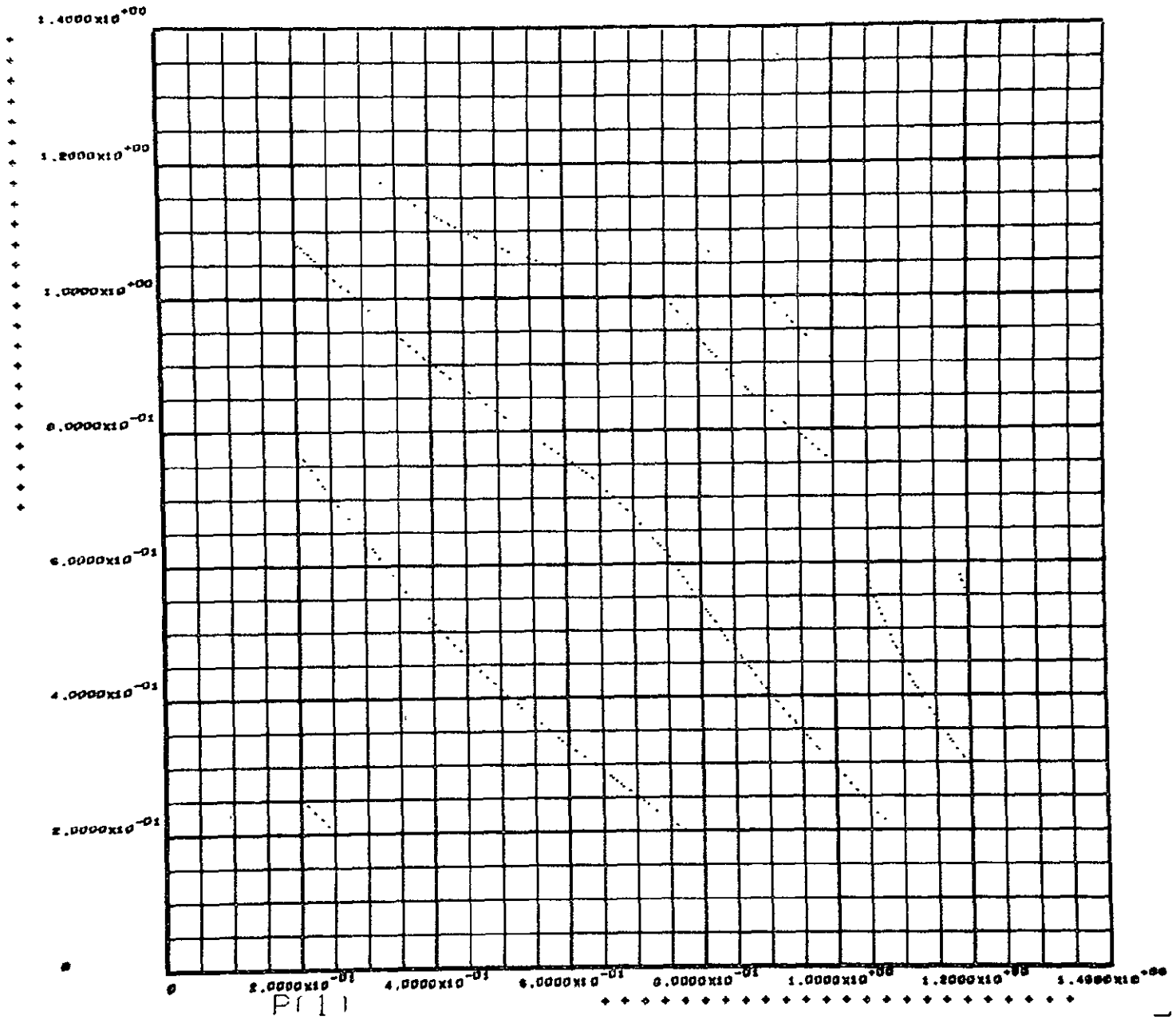
RUN 000000 P. 4 OF SOLUTION 1 1830% 14%



RUN PPL414 P. 3F SOLUTION 1 *****1878*8*****F102*****



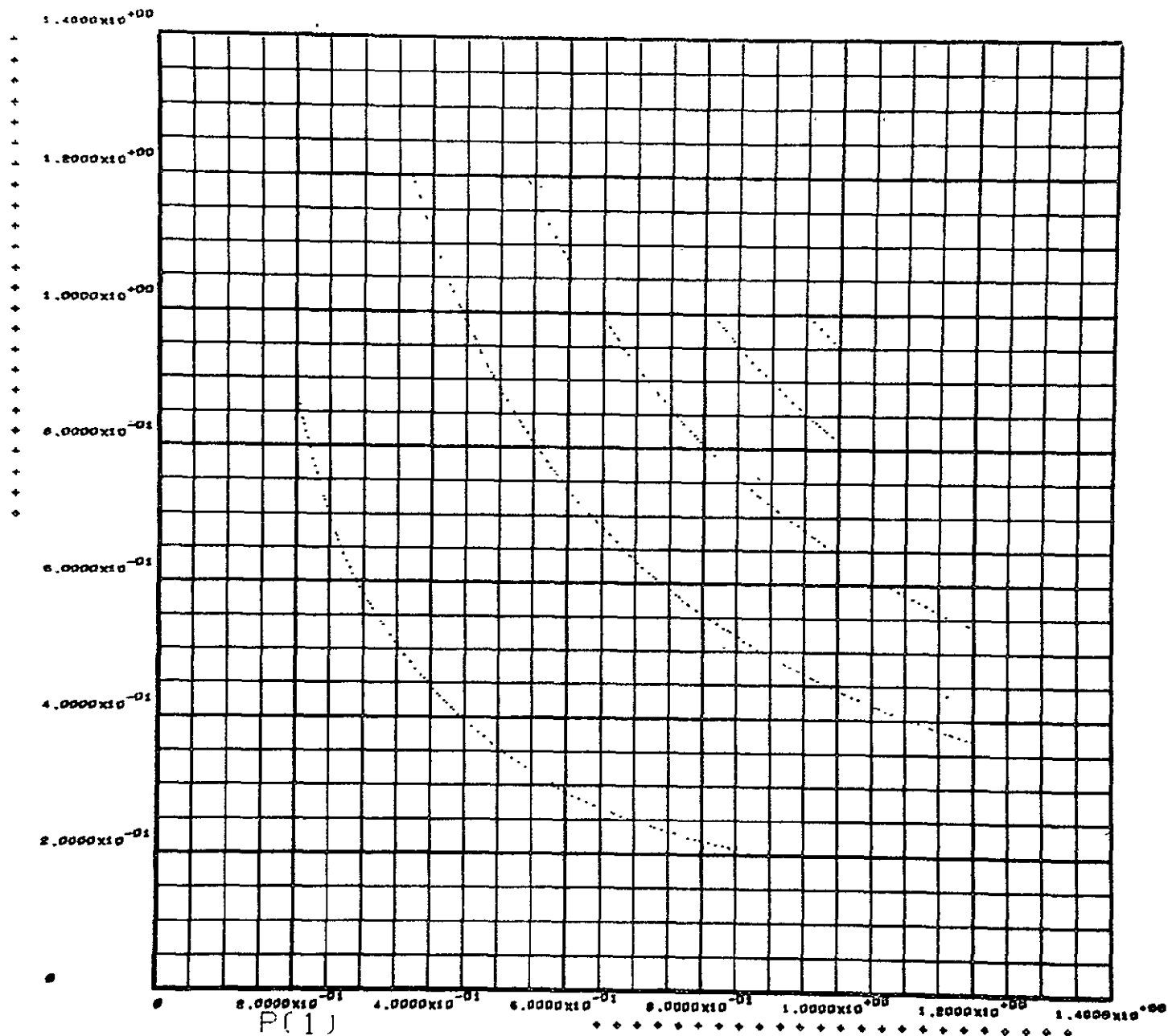
RUN CFL444 H.P. OF SOLUTION 1 ***** 123456789 *****



RUN PPL 414

10 OF 50L TITAN 1

*****123456*****PAGE*****



APPENDIX B

Listings for Simulation
Subroutines and PROC's

BCM11.CONSTR

C TITLE CONSTR - COMPUTE CONSTRAINTS

C AUTHOR P.F. LONG

C DATE 7/17/68

C PURPOSE TO COMPUTE THE DIFFERENCE BETWEEN THE FUNCTIONS AT P AND
C THE TARGET VALUES

C CALL CALL CONSTR(P,CS)

C INPUT BY ARGUMENT LIST

C P INDEPENDENT VARIABLES(PARAMETERS)

C THROUGH COMMON USE

C INP NUMBER OF PARAMETERS

C INC NUMBER OF CONSTRAINTS

C IOP =1, IF OPTIMIZING

C TV(I) TARGET VALUE OF ITH CONSTRAINT FUNCTION

C IFS(I) NUMBER OF THE ITH CONSTRAINT FUNCTION

C OUTPUT CS VALUE OF THE DIFFERENCES

C SUBROUTINE STATEMENT
C SUBROUTINE CONSTR(P,CS)

C SPECIFICATION STATEMENTS

C DIMENSION P(1),CS(1)

C INCLUDE KOMUSE,LIST

C DEFINE FUNCTIONS

C COMMON USE PROVIDES AN INTERFACE BETWEEN ALL USER SUBROUTINES
C AND PAR

C F1(U,V,W,X,Y,Z)=W*X*EXP(Y*Z+U*V)

C F2(U,V,W,X,Y,Z)=10.*W*X*SIN(Y*Z+U*V)

C F3(U,V,W,X,Y,Z)=W*Y**2+5.*X*Z+U*V

C F4(U,V,W,X,Y,Z)=10.*W*X*Y*Z-U*V

C F5(U,V,W,X,Y,Z)=Y**2+Z**2+(1.-W)**2+(1.-X)**2+U**2+V**2

C SET U,V,W,X,Y,Z

C U=P(1)

C V=P(2)

C W=P(3)

C X=P(4)

C Y=P(5)

C Z=P(6)

C COMPUTE THE DIFFERENCE BETWEEN THE FUNCTION AND TARGET VALUE

```
DO 100 J=1,INC
  ICON=IFS(J)
  GO TO (30,40,50,60,70),ICON
30  CS(J)=F1(U,V,W,X,Y,Z)-TV(J)
    GO TO 100
40  CS(J)=F2(U,V,W,X,Y,Z)-TV(J)
    GO TO 100
50  CS(J)=F3(U,V,W,X,Y,Z)-TV(J)
    GO TO 100
60  CS(J)=F4(U,V,W,X,Y,Z)-TV(J)
    GO TO 100
70  CS(J)=F5(U,V,W,X,Y,Z)-TV(J)
100 CONTINUE
    NSE=NS+1
    RETURN
    END
```

BCM11.CHARAC

```
C TITLE          CHARAC-COMPUTE CHARACTERISTICS
C
C AUTHOR          P.F. LONG
C
C DATE            6/18/68
C
C PURPOSE          TO COMPUTE THE CHARACTERISTICS OF A SYSTEM FOR A GIVEN
C                   SET OF VALUES OF THE PARAMETERS
C
C CALL             CALL CHARAC(P,C)
C
C INPUT            BY ARGUMENT LIST
C
C                   P          VECTOR OF PARAMETERS
C
C                   C          VECTOR OF CHARACTERISTICS
C
C SUBROUTINE STATEMENT
C
C                   SUBROUTINE CHARAC(P,C)
C
C SPECIFICATION STATEMENTS
C
C                   DIMENSION P(1),C(1)
C
C DEFINE FUNCTIONS
C
C                   F1(U,V,W,X,Y,Z)=W*X*EXP(Y*Z+U*V)
C                   F2(U,V,W,X,Y,Z)=10.*W*X*SIN(Y*Z+U*V)
C                   F3(U,V,W,X,Y,Z)=W*Y**2+5.*X*Z+U*V
C                   F4(U,V,W,X,Y,Z)=10.*W*X*Y*Z-U*V
C                   F5(U,V,W,X,Y,Z)=Y**2+Z**2+(1.-W)**2+(1.-X)**2+U**2+V**2
C
C SET U,V,W,X,Y,Z
C
C                   U=P(1)
C                   V=P(2)
C                   W=P(3)
C                   X=P(4)
C                   Y=P(5)
C                   Z=P(6)
C
C COMPUTE THE CHARACTERISTICS
C
C                   C(1)=F1(U,V,W,X,Y,Z)
C                   C(2)=F2(U,V,W,X,Y,Z)
C                   C(3)=F3(U,V,W,X,Y,Z)
C                   C(4)=F4(U,V,W,X,Y,Z)
C                   C(5)=F5(U,V,W,X,Y,Z)
C                   RETURN
C                   END
```


BCM11.PRTCHA

C TITLE PRTCHA-PRINT CHARACTERISTICS

C AUTHOR P.F. LONG

C DATE 7/19/68

C PURPOSE TO PRINT THE CHARACTERISTICS AND PARAMETERS FOR A GIVEN
GRID POINT

C CALL CALL PRCHA(P,C)

C INPUT THROUGH CALL LIST

C P VECTOR OF PARAMETERS

C C VECTOR OF CHARACTERISTICS

C OUTPUT PRINTER

C SAME AS INPUT

C SUBROUTINE STATEMENT

C SUBROUTINE PRTCHA(P,C)

C SPECIFICATION STATEMENTS

DIMENSION P(1),C(1)
COMMON/USE/INP,INC,IOP,IFS(5),TV(5)

C PRINT OUTPUTS

WRITE(6,1001)(P(J),J=1,INP)
WRITE(6,1002)(C(J),J=1,5)
RETURN

C FORMAT STATEMENTS

1001 FORMAT(1H0,' P = ',8E15.8/6X,8E15.8)

1002 FORMAT(1H , ' C = ',5E15.8/)

END

PLTCON - P

PAP2*APR26,PAPUSE,PAPUSE

D BY UNIVAC 1108 PDP ON 17 NOV 69 AT 15:38:19 1102-0007

RUNNL PROC

C

C NAMELIST RUNDAT - THESE VARIABLES ARE RELEVANT TO THE RUN BEING
C MADE AND ARE NOT SAVED ON THE SAVE FILES

C

NAMELIST/RUNDAT/KON,ISM,ICM,ISAI,ISAF,ISAC,ISAA,IPRI,
IPRF,IPRC,IPRA,IPRP,IUNIT,NFR,ICHSA,NF14,NF15,NF16,
NF17,NF18

END

SAVNL PROC

C

C NAMELIST SAVDAT - THESE VARIABLES DEFINE THE SIMULATOR AND THE
C ANALYSIS PERFORMED. THIS NAMELIST IS SAVED ON ALL SAVE FILES.

C

NAMELIST/SAVDAT/NP,NC,A,B,DELP,P1I,P2I,NCS,EPS,
NSG,ISGR,NCG,ICCH,IFS,TV

END

RUNDOC PROC

C

C NAMELIST RUNDAT

C

C NAME TYPE/SIZE DEFINITION

C

KON	INT/8	FUNCTION SEQUENCE INDICATOR
ISM	INT/	SEARCH METHOD SELECTOR
ICM	INT/	ITERATIVE METHOD SELECTOR
ISAI	INT/	FLAG,=1 SAVE INISOL OUTPUT
ISAF	INT/	FLAG,=1 SAVE FILGRD OUTPUT
ISAC	INT/	FLAG,=1 SAVE COMCHA OUTPUT
ISAA	INT/	FLAG,=1 SAVE AUXPRO OUTPUT
IPRI	INT/	PRINT LEVEL CONTROL FOR INISOL
IPRF	INT/	PRINT LEVEL CONTROL FOR FILGRD
IPRC	INT/	PRINT LEVEL CONTROL FOR COMCHA
IPRA	INT/	PRINT LEVEL CONTROL FOR AUXPRO
IPRP	INT/	PRINT LEVEL CONTROL FOR PLTCON
IUNIT	INT/	FILE FROM WHICH SAVDAT IS READ
NFR	INT/	NUMBER OF FRAMES TO BE PLOTTED
ICHSA	INT/	FLAG,=1 AFTER READING SAVDAT FROM IUNIT,READ(5,SAVDAT)
NF14	INT/	FLAG,=1 READ PAST SAVDAT ON UNIT 14
NF15	INT/	FLAG,=1 READ PAST SAVDAT ON UNIT 15
NF16	INT/	FLAG,=1 READ PAST SAVDAT ON UNIT 16
NF17	INT/	FLAG,=1 READ PAST SAVDAT ON UNIT 17
NF18	INT/	FLAG,=1 READ PAST SAVDAT ON UNIT 18

END

SAVDOC PROC

C

C NAMELIST SAVDAT

C

C NAME TYPE/SIZE DEFINITION

C

C NP INT/ NUMBER OF INDEPENDENT VARIABLES

PLTCON - P

```

C      NC      INT/      NUMBER OF SYSTEM CHARACTERISTICS
C      A      REA/MP*    LOWER BOUNDS FOR PARAMETERS
C      B      REA/MP*    UPPER BOUNDS FOR PARAMETERS
C      DELP    REA/MP*    STEP SIZE FOR PARAMETERS
C      P1I     REA/      INITIAL VALUE OF GRID PARAMETER 1
C      P2I     REA/      INITIAL VALUE OF GRID PARAMETER 2
C      NCS     INT/      NUMBER OF EQUALITY CONSTRAINTS
C      NTS     INT/      =NCS+1 IF OPTIMIZING,=NCS OTHERWISE
C      EPS     INT/MC*    TOLERANCE ON CONSTRAINTS
C      NSG     INT/      NUMBER OF SOLUTIONS TO BE DEVELOPED
C      ISGR    INT/MG*    ISGR(I)=J, JTH SOLUTION USED TO
C                      START THE ITH GRID
C      NCG     INT/      NUMBER OF GRIDS FOR WHICH
C                      CHARACTERISTICS ARE TO BE COMPUTED
C      ICCH    INT/MH*    ICCH(I)=J, USE THE GRID DEVELOPED FROM
C                      THE JTH SOLUTION TO COMPUTE THE ITH
C                      CHARACTERISTIC SET
C      IFS     INT/5      INDICATES WHICH FUNCTIONS TO USE
C                      AS CONSTRAINTS
C      TV      REA/5      TARGET VALUE FOR THE FUNCTIONS
C
C      * SEE PARAMETER STATEMENT FOR NUMERICAL VALUE
C
KOMUSE      PROC
C
C      COMMON USE PROVIDES AN INTERFACE BETWEEN ALL USER SUBROUTINES
C      AND PAP
C
C      COMMON/USE/INP, INC, IOP, IFS(5), TV(5), NSE

END
INITIL      PROC
C
C      NOI=0
C      NOF=0
C      NOC=0

END
SIMDFN      PROC
C
C      .DEFINE SIMULATOR
C
C      INP=NP
C      INC=NCS

END
RPDDS      PROC
C
C      DIMENSION ISGR(MG), ICCH(MH), A(MP), B(MP),
C      DELP(MP), EPS(MC), IFS(5), TV(5)

END

```

APPENDIX C

Listings of PAP and Subroutines

ARD 4-D PAP TEST

D

APR26.PAP

LEVEL 3

C TITLE PAP - PARAMETRIC ANALYSIS PROGRAM

C AUTHOR P.F. LONG

C DATE 4-3-68

C PURPOSE CONSIDER A SYSTEM WHOSE CHARACTERISTICS ARE CONTINUOUS
 C FUNCTIONS OF THE PARAMETERS P(1), P(2), ..., P(NP). PAP
 C DIRECTS CONTROL TO SUBROUTINES INISOL, FILGRD, COMCHA,
 C AUXPRO, AND PLTCON AS REQUIRED TO PERFORM THE FOLLOWING
 C FUNCTIONS:

C 1. THE PARAMETER SPACE FORMED BY HOLDING P(1) AND P(2)
 C FIXED IS SEARCHED FOR POINTS SATISFYING SPECIFIED
 C CONSTRAINTS.

C 2. AN ATTEMPT IS MADE TO MAP THE SOLUTIONS FOUND TO
 C ALL POINTS OF A GRID FORMED FROM THE DOMAINS OF P(1)
 C AND P(2).

C 3. FOR THE SOLUTIONS FOUND, THE SYSTEM CHARACTERISTICS
 C ARE COMPUTED AND STORED.

C 4. AN INTERFACE IS PROVIDED FOR PROGRAMS DOING
 C ADDITIONAL COMPUTATION BASED ON THE GRIDS DEVELOPED.

C 5. CONTOUR PLOTS OF THE CHARACTERISTICS AND THE
 C NON-GRID PARAMETERS ARE PLOTTED AS SPECIFIED.

C METHOD INPUTS ARE READ WHICH DEFINE THE SIMULATION AND ANALYSES
 C TO BE PERFORMED. THE REQUIRED SUBSEQUENCE OF THE MAJOR
 C SUBROUTINES IS CALLED.

C INPUT

INCLUDE RUNDGC,LIST

INCLUDE SAVDOC,LIST

C NAME TYPE/SIZE DEFINITION

C IOS INT/ PLOTTING OPTION SELECTOR
 C =0, OFF-LINE PLOTS ONLY
 C =1, PRINTER PLOTS ONLY
 C =2, BOTH TYPES

C FFLT HOL/8 48 CHARACTER FRAME TITLE

C ALAB HOL/8 48 CHARACTER ABSCISSA LABEL

C OLAB HOL/8 48 CHARACTER ORDINATE LABEL

C LARC INT/ NOT CURRENTLY USED

C AMI REA/ MINIMUM VALUE OF ABSCISSA

C AMA REA/ MAXIMUM VALUE OF ABSCISSA

C OMI REA/ MINIMUM VALUE OF ORDINATE

C OMA REA/ MAXIMUM VALUE OF ORDINATE

C NFU INT/ NUMBER OF FUNCTIONS ON THIS FRAME

C IUN INT/MU* UNITS WHERE FUNCTIONS ARE TO BE FOUND

C ISO INT/MU* SOLUTION NUMBERS OF FUNCTIONS

C IVA INT/MU* VARIABLE NUMBERS OF FUNCTIONS

ARG 4-D PAP TEST

D

C	ICD	INT/MU*	CONTOUR DIRECTION INDICATOR
C			=0, FUNCTION CONSTANT CONTOURS
C			=1, P1 CONSTANT CONTOURS
C			=2, P2 CONSTANT CONTOURS
C	IAOS	INT/MU*	ABSCISSA-ORDINATE INDICATOR
C			=0, P1/P2, P2/F, OR P1/F IS THE
C			ABSCISSA/ORDINATE
C			=1, THE REVERSE IS TRUE
C	P1IN	INT/MU*	INTERPOLATION INCREMENTS FOR P1
C	P2IN	INT/MU*	INTERPOLATION INCREMENTS FOR P2
C	CHI	REA/MU*	MINIMUM CONTOUR IF IDC=0
C	CMA	REA/MU*	MAXIMUM CONTOUR IF IDC=0
C	CIN	REA/MU*	STEP SIZE BETWEEN CONTOURS IF IDC=0
C	INS	INT/MU*	IF IDC=J, PJ(INS) IS THE FIRST
C			CONTOUR VALUE, J=1 OR 2.
C	EAS	INT/MU*	IF IDC=J, PJ(EAS) IS THE LAST
C			CONTOUR VALUE, J=1 OR 2
C	ISI	INT/MU*	IF IDC=J, PJ(INS+K*ISI), K=0, L-WHERE
C			L*ISI.LE.EAS, ARE THE CONTOUR VALUES
C			PLOTTED, J=1 OR 2

C SUBROUTINES NAME PURPOSE

C	RPD	TO READ PAST NAMELIST SAVDAT ON PLOTFILES
C	OCLOCK	TO GIVE THE TIME
C	INISOL	TO FIND INITIAL SOLUTIONS
C	FILGRD	TO USE INITIAL SOLUTIONS FOUND TO FORM A GRID
C	COMCHA	TO COMPUTE THE SYSTEM CHARACTERISTICS AT EACH
C		GRID POINT
C	AUXPRO	TO PROVIDE AN INTERFACE WITH OTHER PROGRAMS
C	PLTCON	TO PLOT CONTOURS OF THE SYSTEM CHARACTERISTICS

C SIMULATOR THE USER MUST SUPPLY THE FOLLOWING SUBROUTINE ENTRY POINTS:

CONSTR(P,CS) WHICH COMPUTES THE VALUE OF THE
CONSTRAINT FUNCTIONS(CS) AT P.

CHARAC(P,C) WHICH COMPUTES THE CHARACTERISTICS OF
THE SYSTEM (C) AS A FUNCTION OF THE PARAMETERS (P).

PRTCHA(P,C) WHICH PRINTS P AND C AS REQUIRED BY
THE USER.

AUXPRO WHICH PERFORMS OPERATIONS USEFUL TO THE
CURRENT STUDY BUT NOT PROVIDED FOR BY PAP.

IN ADDITION TO THESE SUBROUTINE ENTRY POINTS, THE USER
MUST SUPPLY A PROCEDURE ELEMENT WITH THE FOLLOWING ENTRY
POINTS AND CONTENTS:

RUNDOC CONTAINS THE NAMES AND DEFINITIONS OF THE
VARIABLES IN NAMELIST RUNDAT.

SAVDOC CONTAINS THE NAMES AND DEFINITIONS OF THE
VARIABLES IN NAMELIST SAVDAT.

RD 4-D PAP TEST

DI

C

C

KOMUSE CONTAINS COMMON USE WHICH IS USED FOR COMMUNICATION BETWEEN USER SUBROUTINES.

C

C

RUNNL CONTAINS THE NAMELIST RUNDAT. THIS NAMELIST MUST CONTAIN THOSE INPUTS RELEVANT TO THE CURRENT RUN ONLY. INPUTS KON, IUNIT, ICHSA, AND NFR MUST BE IN THIS NAMELIST. THIS NAMELIST IS NOT SAVED ON THE SAVE FILES.

C

C

C

C

C

C

SAVNL CONTAINS THE NAMELIST SAVDAT WHICH MUST CONTAIN THE INPUTS WHICH DEFINE THE CONSTRAINTS AND THE SIMULATOR. THESE INPUTS WILL BE NEEDED IF FURTHER PROCESSING IS DONE AND ARE WRITTEN ON ALL SAVE FILES. NAMELISTS RUNDAT AND SAVDAT MUST CONTAIN ALL INPUTS REQUIRED TO DEFINE THE SYSTEM, THE CONSTRAINTS TO BE SATISFIED, AND THE INPUTS FOR PAP.

C

C

C

C

C

C

INITIL CONTAINS THE FORTRAN CODE THAT SETS THE VALUES OF ANY VARIABLES THAT MUST BE INITIALIZED AT THE BEGINNING OF EACH PASS. VARIOUS INPUTS WHICH ARE NOT OFTEN CHANGED CAN BE SET HERE. THEIR VALUE CAN BE CHANGED BY THE SUBSEQUENT READ NAMELISTS RUNDAT AND SAVDAT.

C

C

C

C

C

C

SIMDEFN CONTAINS THE FORTRAN CODE WHICH DERIVES FROM THE INPUTS IN RUNDAT AND SAVDAT THE INFORMATION NECESSARY TO DEFINE THE SYSTEM CONFIGURATION, THE CONSTRAINTS TO BE SATISFIED, AND THE INPUTS TO PAP.

C

C

C

C

RPDDS CONTAINS A DIMENSION STATEMENT FOR ALL VARIABLES IN NAMELIST SAVDAT. THE DIMENSION AND SIZE OF THESE VARIABLES MUST CORRESPOND TO THOSE GIVEN TO THE VARIABLES IN PAP.

C

C

C

SPECIFICATION STATEMENTS

C

C

C

INCLUDE PARSTA,LIST
INCLUDE PARDOC,LIST
INCLUDE KOMMGL,LIST
INCLUDE MCLDOC,LIST
INCLUDE KOMUSE,LIST
INCLUDE RUNNL,LIST
INCLUDE SAVNL,LIST
DIMENSION IUN(MU),ISO(MU),IVA(MU),CMI(MU),CMA(MU),
GIN(MU),IGD(MU),TNS(MU),LAS(MU),IST(MU),FTLT(8),ALAB(8),
OLAB(8),IAOS(MU),P1IN(MU),P2IN(MU)
DIMENSION TIME(2)
DATA SUB1/6HINISOL/SUB2/6HFILORD/SUB3/6HCOMCHA/SUB4/
6HAUXPRO/SUB5/6HPLTCON/

C

C

C

C

C

C

C

C

C

C

C

C

C

C

C

NAMELIST PLTDAT-THIS NAMELIST DEFINES ONE PLOTTING FRAME. WHEN SPECIFYING NFR FRAMES, THIS NAMELIST WILL BE READ NFR TIMES IN SEQUENCE STORING THE INPUTS BY

C

NOT REPRODUCIBLE

ARG 4-D PAP TEST

D

C FRAMES.

G

NAMelist/PLTDAT/IOS,IUN,ISO,IVA,FILT,ALAB,OLAB,LABC,
AMI,AMA,OMI,OMA,CMI,CMA,IAOS,P1IN,P2IN,CIN,ICD,INS,LAS,
ISI,NFU
EQUIVALENCE (J,JX)

C

G INITILIZE

C

INCLUDE INITIL,LIST

C

G READ NAMELIST RUNDAT

C

1 READ(5,RUNDAT)

C

G READ NAMELIST SAVDAT FROM FILE IUNIT, 5, OR BOTH (5 IS READ LAST)

C

IF(KON(1).EQ.1) GO TO 10
REWIND IUNIT

READ(IUNIT,SAVDAT)

IF(ICHSA.NE.1) GO TO 20

10

READ(5,SAVDAT)

C

G READ PLOT DATA IF NFR GT 0

C

20 IF(NFR.EQ.0) GO TO 30

C

G READ INPUTS THAT DEFINE A SINGLE FRAME AND STORE IN ARRAYS WHICH
C HAVE ONE OF THEIR SUBSCRIPTS CORRESPONDING TO THE FRAME NUMBER

C

DO 25 J=1,NFR

READ(5,PLTDAT)

IPOP(J)=IOS

LAC(J)=LABC

AMIN(J)=AMI

AMAX(J)=AMA

OMIN(J)=OMI

OMAX(J)=OMA

DO 22 L=1,8

TLT(L,J)=FTLT(L)

ALA(L,J)=ALAB(L)

22

OLA(L,J)=OLAB(L)

NFN(J)=NFU

DO 25 K=1,NFU

IFLU(J,K)=IUN(K)

ISOL(J,K)=ISO(K)

IVAR(J,K)=IVA(K)

IAO(J,K)=IAOS(K)

P1INC(J,K)=P1IN(K)

P2INC(J,K)=P2IN(K)

CMIN(J,K)=CMI(K)

GMAX(J,K)=CMA(K)

DELC(J,K)=CIN(K)

IVS(J,K)=ICD(K)

IS(J,K)=INS(K)

LS(J,K)=LAS(K)

25

INCS(J,K)=ISI(K)

RD 4-D PAP TEST

DA

30 CONTINUE
~~INCLUDE SIMDFN, L-TEST~~

~~C~~
~~C~~ ~~POSITION PLOT FILES BEING USED PAST USRDAT~~
~~C~~

~~IF (NF14.NE.1.OR.NF14.EQ.IUNIT) GO TO 101~~
~~REWIND 14~~

~~CALL RPD(14)~~
101 ~~IF (NF15.NE.1.OR.NF15.EQ.IUNIT) GO TO 102.~~
~~REWIND 15~~

~~CALL RPD(15)~~
~~102~~ ~~IF (NF16.NE.1.OR.NF16.EQ.IUNIT) GO TO 103~~
~~REWIND 16~~

~~CALL RPD(16)~~
103 ~~IF (NF17.NE.1.OR.NF17.EQ.IUNIT) GO TO 104~~
~~REWIND 17~~

~~CALL RPD(17)~~
~~104~~ ~~IF (NF18.NE.1.OR.NF18.EQ.IUNIT) GO TO 105~~
~~REWIND 18~~

~~CALL RPD(18)~~

~~C~~
~~C~~ ~~WRITE SAVDAT ON FILES BEING SAVED THIS PASS~~
~~C~~

~~105~~ ~~IF (ISAI.EQ.0) GO TO 106~~

~~LOI=8+NOI~~

~~NOI=NOI+1~~

~~IF (LOI.GT.10) GO TO 190~~

~~REWIND LOI~~

~~WRITE (LOI, SAVDAT)~~

~~106~~ ~~IF (ISAF.EQ.0) GO TO 107~~

~~LOF=11+NOF~~

~~NOF=NOF+1~~

~~IF (LOF.GT.13) GO TO 190~~

~~REWIND LOF~~

~~WRITE (LOF, SAVDAT)~~

~~107~~ ~~IF (ISAC.EQ.0) GO TO 108~~

~~LOC=14+NOC~~

~~NOC=NOC+1~~

~~IF (LOC.GT.16) GO TO 190~~

~~REWIND LOC~~

~~WRITE (LOC, SAVDAT)~~

~~108~~ ~~IF (ISAA.EQ.0) GO TO 109~~

~~LOA=17+NOA~~

~~NOA=NOA+1~~

~~IF (LOA.GT.18) GO TO 190~~

~~REWIND LOA~~

~~WRITE (LOA, SAVDAT)~~

~~109~~ ~~CONTINUE~~

~~C~~
~~C~~ ~~GO TO THE FUNCTION TO BE PERFORMED NEXT~~
~~C~~

~~J=1~~

~~120~~ ~~KN=KON(J)~~

~~GO TO (130,140,150,160,170),KN~~

~~C~~
~~C~~ ~~CALL INISOL~~
~~C~~

ARD 4-D PAP TEST

DA

```
130      CALL OCLOCK(TIME)
        WRITE(6,1002)SUB1,TIME
        NSE=0
        CALL INISOL
        WRITE(6,1003)NSE,SUB1
        GO TO 180
```

```
C
C      CALL FILGRD
C
```

```
140      CALL OCLOCK(TIME)
        WRITE(6,1002)SUB2,TIME
        NSE=0
        CALL FILGRD
        WRITE(6,1003)NSE,SUB2
        GO TO 180
```

```
C
C      CALL COMCHA
C
```

```
150      CALL OCLOCK(TIME)
        WRITE(6,1002)SUB3,TIME
        CALL COMCHA
        GO TO 180
```

```
C
C      CALL AUXPRO
C
```

```
160      CALL OCLOCK(TIME)
        WRITE(6,1002)SUB4,TIME
        CALL AUXPRO
        GO TO 180
```

```
C
C      CALL PLTCON
C
```

```
170      CALL OCLOCK(TIME)
        WRITE(6,1002)SUB5,TIME
        CALL PLTCON
180      J=J+1
        IF(KON(J)-6) 120,1,200
```

```
C
C      WRITE EXIT REASON MESSAGE
C
```

```
190      WRITE(6,1001) NOI,NOF,NOC,NOA
```

```
C
C      CALL EXIT
C
```

```
200      CALL EXIT
```

```
C
C      FORMAT STATEMENTS
C
```

```
1001      FORMAT(1H , '*****MORE THAN THE ALOTTED NUMBER OF SAVE '
        , 'FILES WERE USED', ' ', NOI = ' ', I3, 2X, 'NOF = ' ', I3, 2X, 'NOC = ' ',
        , I3, 2X, 'NOA = ' ', I3, /)
1002      FORMAT(1H0, '*****CALL ', A6, ' AT TIME ', 2A6, '*****' /)
1003      FORMAT(1H0, '*****SYSTEM EVALUATED ', I4, ' TIMES IN ', A6,
        , '*****' /)
        END
```

PDP-1E APR26.PAPSPC
ED BY UNIVAC 1108 PDP ON 17 SEP 69 AT 08:39:10 1102-0005

KOMSOL PROC

```
C
C SPECIFICATION OF COMMON SOL
C
C COMMON/SOL/NS,SOLP(MP,MS),SOLC(MC,MS)
C
C ...NAME DEFINITION
C
C NS NUMBER OF DISTINCT SOLUTIONS FOUND (.LT.MS)
C SOLP(I,J) ITH COMPONENT OF JTH SOLUTION
C SOLC(I,J) FINAL VALUE OF ITH CONSTRAINT OF JTH SOLUTION
C
C THIS COMMON IS INCLUDED IN SUBROUTINES INISOL AND FILGRD
C
```

END
KOMMCL PROC

```
C
C SPECIFICATION OF COMMON MCL
C
C COMMON/MCL/KON(B),JSM,ICM,ISAI,ISAF,ISAC,ISAA,ISGR(MG),
C ICCH(MH),NFR,NP,NC,A(MP),B(MP),NCS,DELP(MP),
C P1I,P2I,EPS(MC),IPOP(ME),IPRI,IPRF,IPRC,IPRA,IPRP,
C NFN(MF),IFLU(MF,MU),ISOL(MF,MU),NSG,NCG,
C IVAR(ME,MU),TLT(8,ME),ALA(8,ME),OLA(8,ME),LAC(ME),
C AMIN(MF),AMAX(MF),OMIN(MF),OMAX(MF),
C CMIN(ME,MU),CMAX(ME,MU),DELC(ME,MU),IVS(ME,MU),IS(MF,MU),
C LS(MF,MU),INCS(MF,MU),LOI,LOF,LOC,LOA,IUNIT,NOI,NOF,
C NOC,NOA,JX,IAO(ME,MU),P1INC(ME,MU),P2INC(ME,MU)
C
```

END
MCLDOC PROC

```
C
C DEFINITION OF VARIABLES IN MCL
C
C ...VARIABLES THAT CONTROL BOOKKEEPING FUNCTIONS
C
C ...NAME DEFINITION
C
C ISAI FLAG, =1 SAVE OUTPUT FROM INISOL
C ISAF FLAG, =1 SAVE OUTPUT FROM FILGRD
C ISAC FLAG, =1 SAVE OUTPUT FROM COMCHA
C ISAA FLAG, =1 SAVE OUTPUT FROM AUXPRO
C IPRI PRINTOUT LEVEL CONTROLE FOR INISOL
C IPRF PRINTOUT LEVEL CONTROLE FOR FILGRD
C IPRC PRINTOUT LEVEL CONTROLE FOR COMCHA
C IPRA PRINTOUT LEVEL CONTROLE FOR AUXPRO
C IPRP PRINTOUT LEVEL CONTROLE FOR PLTCON
C IUNIT UNIT (OR FILE) NUMBER TO READ INPUT DATA FROM
C LOI OUTPUT UNIT FOR INISOL
C LOF OUTPUT UNIT FOR FILGRD
C LOC OUTPUT UNIT FOR COMCHA
C LOA OUTPUT UNIT FOR AUXPRO
C NOI NUMBER OF PASSES THROUGH INISOL WITH ISAI=1
```

C	NOE	NUMBER OF PASSES THROUGH FILGRD WITH ISAE=1
C	NOC	NUMBER OF PASSES THROUGH COMCHA WITH ISAC=1
C	NOA	NUMBER OF PASSES THROUGH AUXPRO WITH ISAA=1
C	JX	CURRENT VALUE OF KON INDEX

...VARIABLES THAT CONTROL THE ANALYSIS

...	NAME	DEFINITION
-----	------	------------

C	KON(I)	DIRECTS CONTROL
C	ISM	INDICATES SEARCH METHOD TO BE USED
C	ICM	INDICATES ITERATIVE METHOD TO BE USED
C	NP	NUMBER OF INDEPENDENT PARAMETERS
C	NC	NUMBER OF CHARACTERISTICS
C	A(I)	LOWER BOUND ON ITH INDEPENDENT PARAMETER
C	B(I)	UPPER BOUND ON ITH INDEPENDENT PARAMETER
C	DELP(I)	STEP SIZE FOR ITH INDEPENDENT PARAMETER
C	P1I	INITIAL VALUE OF ABSCISSA
C	P2I	INITIAL VALUE OF ORDINATE
C	NCS	NUMBER OF EQUALITY CONSTRAINTS
C	EPS(I)	TOLERANCE ON ITH CONSTRAINT
C	NSG	NUMBER OF SOLUTIONS TO BE DEVELOPED
C	ISGR(J)	=K, KTH SOLUTION USED TO START JTH GRID
C	NCG	NUMBER OF GRIDS TO BE DEVELOPED
C	ICCH(J)	=K, CHARACTERISTICS COMPUTED FOR GRID MAPPED FROM KTH SOLUTION

...VARIABLES THAT CONTROL THE PLOTTING

...	NAME	DEFINITION
-----	------	------------

C	NFR	NUMBER OF PLOTTING FRAMES TO BE GENERATED
C	IPOP(I)	PLOTTING OPTION SELECTOR FOR ITH FRAME
C		=0, OFF-LINE PLOTS ONLY
C		=1, PRINTER PLOTS ONLY
C		=2, BOTH TYPES
C	TLI(J,I)	42 CHARACTER ITH FRAME TITLE
C	ALA(J,I)	48 CHARACTER ITH FRAME ABSCISSA LABEL
C	OLA(J,I)	48 CHARACTER ITH FRAME ORDINATE LABEL
C	LAC(I)	FLAG, =1 LABEL CONTOURS ON ITH FRAME
C	AMIN(I)	MINIMUM VALUE OF ABSCISSA ITH FRAME
C	AMAX(I)	MAXIMUM VALUE OF ABSCISSA ITH FRAME
C	OMIN(I)	MINIMUM VALUE OF ORDINATE ITH FRAME
C	OMAX(I)	MAXIMUM VALUE OF ORDINATE ITH FRAME
C	NFN(I)	NUMBER OF FUNCTIONS TO BE PLOTTED ITH FRAME
C	IFLU(I,J)	FORTRAN LOGICAL UNIT DESIGNATOR FOR JTH FUNCTION OF ITH FRAME
C	ISOL(I,J)	SOLUTION DESIGNATOR FOR JTH FUNCTION OF ITH FRAME
C	IVAR(I,J)	VARIABLE DESIGNATOR FOR JTH FUNCTION OF ITH FRAME
C	P1JNC(I,J)	INTERPOLATION INCREMENT P(1), JTH FUNCTION OF ITH FRAME
C	P2JNC(I,J)	INTERPOLATION INCREMENT P(2), JTH FUNCTION OF ITH FRAME
C	IVS(I,J)	CONTOUR PLANE INDICATOR
C		=0, CONSTANT FUNCTION CONTOURS

C =1, P1 CONSTANT CONTOURS
C =2, P2 CONSTANT CONTOURS
C IAO(I,J) ABSCISSA-ORDINATE INDICATOR
C =0, P1/P2,P2/F, OR P1/F IS THE ABSCISSA/ORDINATE
C =1, THE REVERSE IS TRUE
C CMIN(I,J) MINIMUM CONTOUR JTH FUNCTION ITH FRAME
C CMAX(I,J) MAXIMUM CONTOUR JTH FUNCTION ITH FRAME
C DELC(I,J) STEP SIZE BETWEEN CONTOURS JTH FUNCTION ITH FRAME
C IS(I,J) INITIAL SLICE INDICATOR ITH FRAME JTH FUNCTION
C LS(I,J) FINAL SLICE INDICATOR ITH FRAME JTH FUNCTION
C INCS(I,J) INCREMENT OF SLICES

C THIS COMMON IS INCLUDED IN SUBROUTINES PAP,INISOL,
C FILGRD,COMCHA,AUXPRO,AND PLTCON
C

END
PARSTA PROC

C SPECIFICATION OF PARAMETER STATEMENT

PARAMETER MP=10,MC=8,MF=5,MU=10,MS=25,MG=5,MH=5,MA=31,
MO=MA,MK=1500

END
PARDOC PROC

C DEFINITION OF PARAMETERS

...	NAME	DEFINITION
C	MP	MAXIMUM NUMBER OF INDEPENDENT PARAMETERS
C	MC	MAXIMUM NUMBER OF CONSTRAINTS
C	MF	MAXIMUM NUMBER OF FRAMES
C	MU	MAXIMUM NUMBER OF FUNCTIONS PER FRAME
C	MG	MAXIMUM NUMBER OF GRIDS THAT MAY BE DEVELOPED
C	MS	MAXIMUM NUMBER OF SOLUTIONS THAT MAY BE STORED
C	MH	MAXIMUM NUMBER OF GRIDS FOR WHICH CHARACTERISTICS MAY BE COMPUTED
C	MA	MAXIMUM NUMBER OF ABSCISSAS
C	MO	MAXIMUM NUMBER OF ORDINATES
C	MK	MAXIMUM NUMBER OF CHARACTERISTICS

C PARAMETER STATEMENT INCLUDED IN ALL PAP SUBROUTINES
C

END

NOT REPRODUCIBLE

F APR26.RPD

C TITLE RPD - READ PAST NAMELIST USRDAT

C

C AUTHOR P. F. LONG

C

C DATE 7/26/68

C

C PURPOSE TO POSITION FILE I PAST NAMELIST SAVDAT.

C

C METHOD READ NAMELIST SAVDAT BUT DO NOT TRANSMIT DATA TO PAP.

C

C INPUT THROUGH ARGUMENT LIST

C

C I FILE NUMBER TO BE READ

C

C OUTPUT NONE

C

C SUBROUTINE STATEMENT

C

SUBROUTINE RPD(I)

INCLUDE PARSTA,LIST

INCLUDE RPDDS,LIST

INCLUDE SAVNL,LIST

READ(I,SAVDAT)

RETURN

END

APR26, INISOL

C TITLE INISOL - INITIAL SOLUTION SEARCH SCHEME CONTROL

C

C AUTHOR P.F. LONG

C

C DATE 2-9-68

C

C PURPOSE TO CALL REQUESTED SEARCH SCHEME AND TO PRINT AND STORE
C SOLUTIONS.

C

C CALL CALL INISOL

C

C INPUT THROUGH LABELLED COMMON MCL

C

C ISM SEARCH METHOD SELECTOR

C =J, SEEKJ IS CALLED

C ISAI =1, OUTPUT SAVED ON UNIT LOI

C LOI OUTPUT UNIT NUMBER

C

C A,B,DELP,P1I,P2I,NP,NCS,ICM,EPS,AND IPRI ARE PASSED ON
C TO THE VARIOUS SEARCH SUBROUTINES.

C THESE VARIABLES ARE DEFINED IN THE MAIN PROGRAM UNDER
C COMMON MCL.

C

C OUTPUT NS,SOLP, AND SOLC ARE PRINTED,STORED IN COMMON SOL, AND
C IF ISAI=1, WRITTEN ON UNIT LOI.

C THESE VARIABLES ARE DEFINED BELOW UNDER COMMON SOL.

C

C SUBROUTINES SEEK1,SEEK2,SEEK3,SEEK4,SEEK5

C

C SUBROUTINE STATEMENT

C

SUBROUTINE INTSOL

C

C SPECIFICATION STATEMENTS

C

INCLUDE PARSTA,LIST

INCLUDE KOMMCL,LIST

INCLUDE KOMSOL,LIST

C

C CALL THE SEARCH METHOD INDICATED BY ISM TO FIND THE INITIAL
C SOLUTIONS

C

GO TO(5,10,15,20,25),ISM

5 CALL SEEK1(A,B,DELP,P1I,P2I,NP,NCS,ICM,EPS,NS,SOLP,SOLC,
IPRI)

GO TO 30

10 CALL SEEK2(A,B,DELP,P1I,P2I,NP,NCS,ICM,EPS,NS,SOLP,SOLC,
IPRI)

GO TO 30

15 CALL SEEK3(A,B,DELP,P1I,P2I,NP,NCS,ICM,EPS,NS,SOLP,SOLC,
IPRI)

GO TO 30

20 CALL SEEK4(A,B,DELP,P1I,P2I,NP,NCS,ICM,EPS,NS,SOLP,SOLC,
IPRI)

GO TO 30

25 CALL SEEK5(A,B,DELP,P1I,P2I,NP,NCS,ICM,EPS,NS,SOLP,SOLC,
IPRI)

C
C WRITE SOLUTIONS ON LOI
C

30 IF (ISAI.NE.1) GO TO 35
WRITE(LOI) NS,SOLP,SOLC
END FILE LOI
REWIND LOI

C
C PRINT SOLUTIONS
C

35 WRITE(6,1001) NS
DO 40 J=1,NS
40 WRITE(6,1002) J,(SOLP(I,J),I=1,NP)
RETURN

C
C FORMAT STATEMENTS
C

1001 FORMAT(1H0,4X,I2,' SOLUTION(S) FOUND.'/)
1002 FORMAT(1H,8X,I2,(8E15.8/11X))
END

APR26.FILGRD

C TITLE FILGRD - SOLUTION MAPPING CONTROL

C AUTHOR P.F. LONG

C DATE 3-7-68

C PURPOSE TO CALL GRIDMK WITH EACH INITIAL SOLUTION REQUESTED AND
STORE THE RESULTS ON UNIT LOF.

C METHOD LOAD THE REQUESTED SOLUTIONS AND CALL GRIDMK FOR
SOLUTION MAPPING.

C CALL CALL FILGRD

C INPUTS THROUGH LABELLED COMMON MCL

C KON,IUNIT,NSG,ISAF,LOF,A,B,DELP,ISGR,EPS,NP,NCS,ICM,IPRF
C THESE VARIABLES ARE DEFINED IN THE MAIN PROGRAM UNDER
C COMMON MCL.

C THROUGH LABELLED COMMON SOL

C NS,SOLP,SOLC THESE VARIABLES ARE DEFINED BELOW UNDER
C COMMON SOL.

C OUTPUT ON UNIT LOF

C NP1 NUMBER OF P(1)'S

C NP2 NUMBER OF P(2)'S

C NP NUMBER OF PARAMETERS

C ISGR(K) NUMBER OF SOLUTION FROM WHICH GRID WAS MAPPED

C GRD(I,J,K) RESULTING SOLUTION GRID. I=1,NP J=1,NP1
K=1,NP2

C COPT(J,K) OPTIMUM VALUE IF OPTIMIZING. J=1,1,NP1,K=1,NP2

C IDCX(J,K) =1, SOLUTION FOUND FOR GRID POINT CORRESPONDING
C TO (J,K)

C SUBROUTINES GRIDMK

C SUBROUTINE STATEMENT

C SUBROUTINE FILGRD

C SPECIFICATION STATEMENTS

C INCLUDE PARSTA,LIST

C INCLUDE KOMMCL,LIST

C INCLUDE KOMSOL,LIST

C DIMENSION GRD(MP,MA,MO),COPT(MA,MO),PINI(MP),IDCV(MA,MO)

C IF FILGRD IS THE FIRST MAJOR SUBROUTINE CALLED THIS PASS, READ
C NS,SOLP,SOLC FROM IUNIT

C IF(KON(1).NE.2)GO TO 10

C READ(IUNIT) NS,SOLP,SOLC

C IF(NS.LT.NSG) NSG=NS

```
C
C   DESIGNATE STORAGE DEVICE
C
C       IF (ISAF.NE.1) LOF=25
C
C   COMPUTE NUMBER OF P(1)'S AND P(2)'S
C
C       NP1=(B(1)-A(1))/DELP(1)+1.01
C       NP2=(B(2)-A(2))/DELP(2)+1.01
C
C   CALL GRIDMK TO GENERATE THE GRIDS FOR THE INDICATED SOLUTIONS
C
C       DO 100 K=1,NSG
C
C   ...SET INITIAL SOLUTION
C
C       JJ=ISGR(K)
C       DO 20 J=1,NP
C   20      PINI(J)=SOLP(J,JJ)
C
C   ...IF JJ .GT. NS WRITE ERROR MESSAGE AND RETURN
C
C       IF (JJ.GT.NS) GO TO 105
C
C   ...CALL GRIDMK TO DEVELOP THE GRID FOR THIS SOLUTION
C
C       WRITE(6,1002) JJ
C       CALL GRIDMK(A,B,DELP,EP,S,PINI,NP,NCS,ICM,GRD,COPT,IDCV,
C       IPRF)
C
C   WRITE GRIDS AND OPTIMUM VALUES ON UNIT LOF
C
C   50      WRITE(LOF) NP1,NP2,NP,ISGR(K),GRD,COPT,IDCV
C   100     CONTINUE
C          REWIND LOF
C          IF (LOF.NE.25.AND.KON(JX+1).EQ.3) CALL RPD(LOF)
C   102     RETURN
C
C   WRITE ERROR MESSAGE
C
C   105     WRITE(6,1001) JJ
C          RETURN
C
C   FORMAT STATEMENTS
C
C   1001     FORMAT (1H '*****SOLUTION NUMBER ',I2,' NOT AVAILABLE')
C   1002     FORMAT(1H0,'*****MAP SOLUTION NO. ',I2,'*****'/)
C          END
```

APR26.COMCHA

```
C TITLE          COMCHA-COMPUTE SYSTEM CHARACTERISTICS
C
C AUTHOR          P.F. LONG
C
C DATE            6-7-68
C
C PURPOSE         TO COMPUTE THE SYSTEM CHARACTERISTICS FOR EACH POINT AND
C                 STORE FOR PLOTTING.
C
C METHOD           THE INDICATED GRID AND THE CORRESPONDING SOLUTIONS ARE
C                 READ AND THE CHARACTERISTICS ARE COMPUTED AND STORED FOR
C                 EACH POINT. PRCHA IS CALLED FOR ALL OR A SUBSET OF THE
C                 POINTS TO PRINT THE SYSTEM PARAMETERS AND CHARACTERISTICS
C
C INPUT           THROUGH LABELLED COMMON MCL
C
C                 KON,IUNIT,ISAC,NCG,ICCH,IPRC,NP,NC
C                 THESE VARIABLES ARE DEFINED IN THE MAIN PROGRAM UNDER
C                 COMMON MCL.
C
C                 READ FROM UNIT LIC
C
C                 NP1      NUMBER OF P(1)'S
C                 NP2      NUMBER OF P(2)'S
C                 NP       NUMBER OF PARAMETERS
C                 ISNO     NUMBER OF SOLUTION FROM WHICH GRID WAS MAPPED
C                 GRD(I,J,K) RESULTING SOLUTION GRID. I=1,NP J=1,NP1
C                           K=1,NP2
C                 COPT(J,K) OPTIMUM VALUE IF OPTIMIZING. J=1,1,NP1,K=1,NP2
C                 IDCX(J,K) =1, SOLUTION FOUND FOR GRID POINT CORRESPONDING
C                           TO (J,K)
C
C OUTPUT          PARAMETERS AND CHARACTERISTICS ARE PRINTED AND WRITTEN
C                 ON UNIT LOC
C
C SUBROUTINES     CHARAC  COMPUTES THE SYSTEM CHARACTERISTICS GIVEN THE
C                       VALUES OF THE PARAMETERS
C                       PRCHA PRINTS PARAMETERS AND CHARACTERISTICS FOR A
C                       SINGLE GRID POINT
C
C SUBROUTINE STATEMENT
C
C SUBROUTINE COMCHA
C
C SPECIFICATION STATEMENTS
C
C                 INCLUDE PARSTA,LIST
C                 INCLUDE KOMMCL,LIST
C                 DIMENSION GRD(MP,MA,MO),COPT(MA,MO),P(MP),IDCV(MA,MO),
C                 C(MK)
C
C DETERMINE INPUT AND OUTPUT UNITS
C
C                 LIC=25
C                 IF(KON(1).EQ.3) LIC=IUNIT
C                 IF(ISAC.EQ.0) LOC=26
```

```

C
C   DEVELOP THE GRIDS
C
C       DO 200 I=1,NCG
C
C   ...READ A GRID FROM FLU LIC
C
C       READ(LIC)NP1,NP2,NP,ISNO,GRD,COPT,IDCV
C
C   ...IS THIS GRID TO BE DEVELOPED
C
C       IF(ICCH(I)-ISNO)20,15,200
C
C   ...WRITE ERROR MESSAGE AND CONTINUE
C
20       WRITE(6,1001)
        GO TO 200
C
C   ....WRITE IDENTIFYING INFORMATION ON LEU LOC
C
15       WRITE(LOC)ISNO,NP1,NP2,NP,NC
        WRITE(6,1002) ISNO
C
C   ...READ, COMPUTE, AND PRINT GRID POINT BY GRID POINT
C
C       DO 190 J=1,NP1
C       DO 190 K=1,NP2
C
C   ...SET PARAMETERS FOR THIS GRID POINT
C
C       DO 30 L=1,NP
30       P(L)=GRD(L,J,K)
        IF(IDCV(J,K).NE.0) GO TO 40
        P(1)=A(1)+(J-1)*DELP(1)
        P(2)=A(2)+(K-1)*DELP(2)
        DO 35 K1=3,NP
35       P(K1)=0.0
        DO 36 K1=1,NC
36       C(K1)=0.0
C
C   ....IF A SOLUTION WAS FOUND FOR THIS POINT COMPUTE CHARACTERISTICS
C
C       IF(IDCV(J,K).EQ.1) CALL CHARAC(P,C)
C
C       WRITE P AND C ON PLOT TAPE
C
C       WRITE(LOC)(IDCV(J,K),(P(I),I=1,NP),(C(JJ),JJ=1,NC))
C
C   ...IF THIS POINT IS TO BE PRINTED CALL PRITCHA
C
C       IF(IPRC.EQ.0) GO TO 190
C       IF(IDCV(J,K).NE.1) GO TO 190
C       IF(MOD(J,IPRC).EQ.0.AND.MOD(K,IPRC).EQ.0.OR.J.EQ.1.OR.K
C       .EQ.1.OR.J.EQ.NP1.OR.K.EQ.NP2) CALL PRITCHA(P,C)
190      CONTINUE
200      CONTINUE
        END FILE LOC

```

REWIND LOC

REWIND 25

RETURN

205

C

C

EORMAT STATEMENTS

C

1001

FORMAT(1H0,'***** GRID REQUESTED WAS NOT MAPPED.*****')

1002

FORMAT(1H0,'*****COMPUTE CHARACTERISTICS FOR SOLUTION '

'NO. ',I2,'*****'/)

END

APR26.PLITCON

C TITLE PLITCON - CONTOUR PLOTTING

C

C AUTHOR H.T. HINMAN

C

C DATE 1/1/69

C

C PURPOSE TO READ FUNCTIONS OF TWO VARIABLES FROM A NUMBER OF
C FILES AND TO GENERATE AND PLOT CONTOURS OF THESE
C FUNCTIONS

C

C METHOD PLITCON DETERMINES IN ADVANCE WHAT FUNCTIONS ARE TO BE
C READ SO THAT EACH FILE CAN BE READ IN ONE PASS. EACH
C FUNCTION IS THEN STORED ON A SEPARATE FASTRDUM FILE.
C PLITCON THEN CALLS FRMPLT WHICH PROVIDES FOR THE CONTOUR
C GENERATION AND PLOTTING.

C

C INPUT VIA COMMON MCL. SEE DOCUMENTATION OF MCL

C

C OUTPUT SUMMARY TABLES, FUNCTION ARRAYS, PRINTER PLOTS, PLOT TAPE

C

C

C SUBROUTINE STATEMENT

C

.. SUBROUTINE PLITCON

C

C SPECIFICATION STATEMENTS

C

INCLUDE PARSTA,LIST

INCLUDE KOMMCL,LIST

INCLUDE MCLDOC,LIST

INCLUDE KPFILE,LIST

PARAMETER MEU=ME*MU, MPC=NP+MK

INTEGER SKIP

DIMENSION MIFLU(MEU), MISOL(MEU), MIVAR(MEU), EAINC(MEU),

• FOINC(MEU), FCMIN(MEU), FCMAX(MEU), FDELC(MEU), MIVS(MEU), MIS(MEU),

• MLS(MEU), MAINCS(MEU), EAMIN(MEU), EAMAX(MEU), EOMIN(MEU), EOMAX(MEU),

• INSA(MEU), IFASTD(MEU), MIPOP(MEU)

DIMENSION LHVAR(50), LHCT(50), LHNH(50), HP1(MA), HP2(MO),

• LHFLU(50), LHSOL(50)

DIMENSION V(MPC), P(2)

C

C SET NON-APPLICABLE STORAGE AREAS EQUAL TO ZERO

C

DO 90 J=1, NFR

JP = NFN(I)

DO 90 J=1, JP

IF (IVS(I,J).GT.0) GO TO 80

IVS(I,J) = 0

IS(I,J) = 0

LS(I,J) = 0

INCS(I,J) = 0

GO TO 90

80 IF (IVS(I,J).EQ.1) P1INC(I,J)=0

IF (IVS(I,J).EQ.2) P2INC(I,J)=0

CMIN(I,J) = 0

CMAX(I,J) = 0

```

      DELC(I,J) = 0
90      CONTINUE
C
C      IDRUM = THE FIRST FASTDRUM NUMBER TO BE USED FOR STORAGE
C
      IDRUM = 31
      NTFN = 0
C
C      WRITE A FRAME BY FRAME SUMMARY OF PLOTTING TO BE DONE
C
      IF (IPRP.LT.1) GO TO 150
      WRITE(6,5)
      WRITE(6,6)
      DO 120 I=1,NFR
      WRITE(6,7) I,IPOP(I),AMIN(I),AMAX(I),OMIN(I),OMAX(I)
      JL = NFN(I)
      DO 120 J = 1,JL
120      WRITE(6,8) IFLU(I,J),ISOL(I,J),IVAR(I,J),IAO(I,J),
      P1INC(I,J),P2INC(I,J),CMIN(I,J),CMAX(I,J),DELC(I,J),
      IVS(I,J),IS(I,J),LS(I,J),INCS(I,J)
150      CONTINUE
C
C      CALCULATE NUMBER OF VARIABLE REQUESTS
C
      DO 110 I=1,NFR
110      NTFN = NTFN+NFN(I)
C
C      SET FASTDRUM FILE INDICATORS TO INITIAL CONDITION
C
      DO 130 I=1,100
      NIPOP(I) = 0
130      NFORM(I) = 0
      DO 140 I=1,MFU
140      IFASTD(I)=0
      ICT = 0
C
C      RESTORE VARIABLES FOR SORTING
C
      DO 170 I=1,NFR
      JP = NFN(I)
      DO 160 J=1,JP
      ICT = ICT+1
      MIPOP(ICT) = IPOP(I)
      MIFLU(ICT) = IFLU(I,J)
      MISOL(ICT) = ISOL(I,J)
      MIVAR(ICT) = IVAR(I,J)
      FAINC(ICT) = P1INC(I,J)
      FOINC(ICT) = P2INC(I,J)
      FCMIN(ICT) = CMIN(I,J)
      FCMAX(ICT) = CMAX(I,J)
      EDELC(ICT) = DELC(I,J)
      MIVS(ICT) = IVS(I,J)
      MIS(ICT) = IS(I,J)
      MLS(ICT) = LS(I,J)
      MINCS(ICT) = INCS(I,J)
      IF(IAO(I,J).EQ.1) GO TO 155
      FAMIN(ICT) = AMIN(I)
```

```

      FOMAX(ICT) = AMAX(I)
      FOMIN(ICT) = OMIN(I)
      FOMAX(ICT) = OMAX(I)
      GO TO 157
155      FAMIN(ICT) = OMIN(I)
      FOMAX(ICT) = OMAX(I)
      FOMIN(ICT) = AMIN(I)
      FOMAX(ICT) = AMAX(I)
157      IF (IVS(I,J) .NE. 2) GO TO 160
      FOMIN(ICT) = FAMIN(ICT)
      FOMAX(ICT) = FAMAX(ICT)
160      INSA(ICT) = ICT
170      CONTINUE
C
C      SORT UNIT NUMBER, SOLUTION NUMBER, AND VARIABLE NUMBER IN
C      ASCENDING ORDER
C
      NUMRAY = 18
      NUMELM = ICT
      CALL ORDUP(NUMRAY, NUMELM, MIVAR, MIFLU, MISOL, FAINC, FOINC,
      • EDMIN, ECMAX, EDEL, MIVS, MIS, MLS, MINCS, FAMIN, FAMAX, FOMIN, FOMAX,
      • MIPOP, INSA, I, J)
      CALL ORDUP(NUMRAY, NUMELM, MISOL, MIFLU, MIVAR, EAINC, EOINC,
      • FDMIN, FCMAX, FDEL, MIVS, MIS, MLS, MINCS, FAMIN, FAMAX, FOMIN, FOMAX,
      • MIPOP, INSA, I, J)
      CALL ORDUP(NUMRAY, NUMELM, MIFLU, MISOL, MIVAR, FAINC, FOINC,
      • EDMIN, ECMAX, EDEL, MIVS, MIS, MLS, MINCS, FAMIN, FAMAX, FOMIN, FOMAX,
      • MIPOP, INSA, I, J)
C
C      STORE EACH UNIQUE SOLUTION VARIABLE REQUESTED IN A SEPARATE AREA
C      AND CHECK NUMBER OF REQUESTS FOR EACH SOLUTION VARIABLE
C
      KFC = 1
      LHFLU(1) = MIFLU(1)
      LHSOL(1) = MISOL(1)
      LHVAR(1) = MIVAR(1)
      M1 = 2
      M2 = NTFN
280      DO 300 IL=M1, M2
      IF (LHFLU(KFC).EQ.MIFLU(IL).AND.LHSOL(KFC).EQ.MISOL(IL)
      • .AND.LHVAR(KFC).EQ.MIVAR(IL)) GO TO 300
      GO TO 310
300      CONTINUE
      LHCT(KFC) = M2-M1+2
      GO TO 330
310      LHCT(KFC) = IL-M1+1
      M1 = IL+1
      KFC = KFC+1
      LHFLU(KFC) = MIFLU(IL)
      LHSOL(KFC) = MISOL(IL)
      LHVAR(KFC) = MIVAR(IL)
      IF (IL.LT.M2) GO TO 280
      LHCT(KFC) = 1
C
C      CHECK HOW MANY DIFFERENT REQUESTS FOR EACH UNIQUE VARIABLE
C      STORE THE NUMBER OF THE FASTORUM FILE WITH EACH REQUEST AND
C      STORE THE NUMBER OF DIFFERENT REQUESTS (THE NUMBER OF FILES TO BE

```


C CREATED) FOR EACH UNIQUE VARIABLE

```
C
330      IJ2 = 0
          NAD = 1
          NDR = IDRUM
335      IJ = IJ2+1
          IJ2 = IJ2+LHGT(MAD)
345      IJ1 = IJ+1
          IEASTD(IJ) = NDR
          IPOPS = MIPOP(IJ)
          IF(IPOPS.NE.0)    NIPOP(NDR)=1
          IF(IJ1.GT.IJ2)    GO TO 500
          DO 400    I=IJ1,IJ2
              IF(ABS(FAINC(IJ)-FAINC(I)).GT.1.0E-10)    GO TO 400
              IF(ABS(FOINC(IJ)-FOINC(I)).GT.1.0E-10)    GO TO 400
              IF(ABS(FDMIN(IJ)-FDMIN(I)).GT.1.0E-10)    GO TO 400
              IF(ABS(ECMAX(IJ)-ECMAX(I)).GT.1.0E-10)    GO TO 400
              IF(ABS(FDELC(IJ)-FDELC(I)).GT.1.0E-10)    GO TO 400
              IF(ABS(MIVS(IJ)-MIVS(I)).GT.1.0E-10)    GO TO 400
              IF(ABS(MIS(IJ)-MIS(I)).GT.1.0E-10)    GO TO 400
              IF(ABS(MLS(IJ)-MLS(I)).GT.1.0E-10)    GO TO 400
              IF(ABS(MINCS(IJ)-MINCS(I)).GT.1.0E-10)    GO TO 400
          IEASTD(I) = NDR
          NFORM(NDR) = 1
          IPOPS = IPOPS+MIPOP(I)
400      CONTINUE
          IF(IPOPS.NE.0)    NIPOP(NDR)=1
          DO 450    I=IJ1,IJ2
              IF(IEASTD(I).EQ.0)    GO TO 470
450      CONTINUE
          GO TO 500
470      NDR = NDR+1
          IJ = I
          GO TO 345
500      LHNR(NAD) = NDR
          IF(NAD.EQ.KFC)    GO TO 550
          NDR = NDR+1
          NAD = NAD+1
          GO TO 335
```

C
C READ TAPE

```
C
550      NAD = 1
          IF(IPRP.LT.1) GO TO 555
          WRITE(6,1)
555      NFD1 = IDRUM
560      NT = LHELU(NAD)
570      READ(NT) NHSOL,NA,NO,NP,NC
          IF(IPRP.LT.1) GO TO 575
          WRITE(6,2) NT,NHSOL
575      NTF = NA*NO
          IF(NHSOL.EQ.LHSOL(NAD))    GO TO 600
```

C
C IF SOLUTION NUMBERS DO NOT MATCH, SKIP REST OF FILE

```
C
          DO 580    I=1,NTF
580      READ(NT) SKIP
```

GO TO 570

C
C ~~WRITE HEADER INFORMATION ON EACH FASTDRUM FILE NOW BEING PROCESSED~~
C

600 IV = LHVAR(NAD)
NFD2 = LHNR(NAD)
NPC=NP+NC-2
DO 602 ID = NFD1,NFD2
WRITE(ID) NT,NHSOL,IV,NA,NO
IF(IPRP.LT.1) GO TO 602
WRITE(6,3) ID,NHSOL,IV
REWIND ID
602 READ(ID) NT,NHSOL,IV,NA,NO
CONTINUE
IF (NAD.EQ.KFC) GO TO 610
NAD1 = NAD+1
DO 608 I=NAD1,KFC
IF(LHFLU(NAD).EQ.LHFLU(I).AND.LHSOL(NAD).EQ.LHSOL(I))
GO TO 605
GO TO 610
605 NFD3 = LHNR(I-1)+1
NFD2 = LHNR(I)
IV = LHVAR(I)
DO 608 ID = NFD3,NFD2
WRITE(ID) NT,NHSOL,IV,NA,NO
IF(IPRP.LT.1) GO TO 608
WRITE(6,3) ID,NHSOL,IV
608 CONTINUE
610 DO 710 ITA=1,NA
DO 700 ITO=1,NO
READ(NI) IDCX,P(1),P(2),(V(JJ),JJ=1,NPC)

C
C ~~WRITE FASTDRUM RECORD FOR EACH UNIQUE VARIABLE REQUEST~~
C

HP2(ITO) = P(2)
IV = LHVAR(NAD)
NFD2 = LHNR(NAD)
DO 650 ID=NFD1,NFD2
650 WRITE(ID) IDCX,V(IV)
IF(NAD.EQ.KFC) GO TO 700
DO 670 I=NAD1,KFC
IF(LHFLU(NAD).EQ.LHFLU(I).AND.LHSOL(NAD).EQ.LHSOL(I))
GO TO 660
GO TO 700
660 NFD3 = LHNR(I-1)+1
IV = LHVAR(I)
NFD2 = LHNR(I)

C
C ~~WRITE FASTDRUM RECORDS FOR EVERY OTHER UNIQUE VARIABLE REQUEST~~
C ~~FOR THE SAME SOLUTION~~
C

DO 665 ID = NFD3,NFD2
665 WRITE(ID) IDCX,V(IV)
670 CONTINUE
700 CONTINUE
710 HP1(ITA) = P(1)

C

C WRITE GRID PARAMETERS FOR EACH FASTDRUM FILE

```

C
      NFD2 = LHNH(NAD)
      DO 720 ID=NFD1,NFD2
720    WRITE(ID) (HP1(I),I=1,NA),(HP2(J),J=1,NO)
      IF(NAD.EQ.KFC) GO TO 800
      DO 740 I=NAD1,KFC
      IF(LHFLU(NAD).EQ.LHFLU(I).AND.LHSOL(NAD).EQ.LHSOL(I))
      GO TO 735
      GO TO 800
735    NFD3 = LHNH(I-1)+1
      NFD2 = LHNH(I)
      DO 740 ID=NFD3,NFD2
740    WRITE(ID) (HP1(K),K=1,NA),(HP2(J),J=1,NO)
800    IF(NAD.EQ.KFC) GO TO 900
      DO 810 I=NAD1,KFC
      IF(LHFLU(NAD).EQ.LHFLU(I).AND.LHSOL(NAD).EQ.LHSOL(I))
      GO TO 810
      GO TO 820
810    CONTINUE
      GO TO 900
820    NAD = I
      NED1 = LHNH(I-1)+1
      GO TO 560
900    CONTINUE

```

C
C REORDER AND STORE FASTDRUM FILE NUMBER INDICATOR FOR EACH
C FUNCTION REQUEST
C

```

      ICT = 0
      CALL ORDUP(2,NUMELM,INSA,IFASTD,I1,I2,I3,I4,I5,I6,I7,I8,
      I9,I10,I11,I12,I13,I14,I15,I16,I17,I18)
      DO 920 I=1,NFR
      JP = NFN(I)
      DO 920 J=1,JP
      ICT = ICT+1
920    IFD(I,J) = IFASTD(ICT)

```

C
C REWIND FILES
C

```

      DO 950 N=1,NPRIM,INDP
      REWIND N
      READ(N).MT,NHSOL,IV,NA,NO
950    REWIND N

```

C
C CALL FRMPLT FOR CONTOUR GENERATION AND PLOTTING
C

```

      CALL FRMPLT
      RETURN

```

C
C FORMAT STATEMENTS
C

```

1    FORMAT(1H1)
2    FORMAT(32H0PRESENT SOLUTION NUMBER ON UNIT,I3,5H IS,I3)
3    FORMAT(23H FASTDRUM FILE,I3,26H CONTAINS SOLUTI
      .ON NUMBER,I3,21H AND VARIABLE NUMBER,I3)
5    FORMAT(60H1FRAME NO. PLOT OP. AMIN AMAX 0

```

```
      .MIN      .OMAX)
6      FORMAT(1H0,29X,103HUNIT SOL.NO. VAR.NO. XY OP. DELP
      .1 DELP2 CMIN CMAX CDEL IVS IS LS INCS/
      ./)
7      FORMAT(1H0,I7,I10,4X,4F10.1/)
8      FORMAT(29X,I4,3I8,2X,5F10.2,2X,4I5)
      END
```

APR26.SEEK1

C TITLE SEEK1 - TOTAL DOMAIN SEARCH

C AUTHOR P.F. LONG

C DATE 2-14-68

C PURPOSE TO FIND ALL DISTINCT SOLUTIONS WITHIN THE GIVEN
PARAMETER SPACE.

C METHOD START AN ITERATIVE PROCEDURE AT EACH POINT OF A GRID
FORMED FROM THE DOMAINS OF P(3),P(4),...,P(NA). RETURN
THE RESULTING DISTINCT SOLUTIONS.

C CALL SEEK1(A,B,DELP,P1I,P2I,MP,NCS,ICM,EPS,NS,SOL,SOLC,IPRI)

C INPUT THROUGH ARGUMENT LIST

C NAME DEFINITION

C A(I) LOWER BOUND ON ITH INDEPENDENT PARAMETER

C B(I) UPPER BOUND ON ITH INDEPENDENT PARAMETER

C DELP(I) STEP SIZE FOR ITH INDEPENDENT PARAMETER

C P1I VALUE OF P(1) WHILE SEARCHING

C P2I VALUE OF P(2) WHILE SEARCHING

C NP NUMBER OF INDEPENDENT PARAMETERS

C NCS NUMBER OF EQUALITY CONSTRAINTS

C ICM ITERATIVE METHOD SELECTOR

C EPS(I) TOLERANCE ON ITH EQUALITY CONSTRAINT

C IPRI PRINT CONTROL

C OUTPUT THROUGH ARGUMENT LIST

C NAME DEFINITION

C NS NUMBER OF DISTINCT SOLUTIONS FOUND

C SOLP(I,J) ITH COMPONENT OF JTH SOLUTION

C SOLC(I,J) FINAL VALUE OF ITH CONSTRAINT OF JTH SOLUTION

C SUBROUTINES CONV

C SUBROUTINE STATEMENT

C SUBROUTINE SEEK1(A,B,DELP,P1I,P2I,MP,NCS,ICM,EPS,NS,
SOLP,SOLC,IPRI)

C SPECIFICATION STATEMENTS

C INCLUDE PARSTA,LIST

C DIMENSION A(1),B(1),DELP(1),EPS(1),SOLP(MP,MS),

SOLC(IC,MS),PI(MP),PE(MP),CSE(MC)

C DIMENSION TYCO(5)

C DATA TYCO/6HCONVER,6HNEASOL,6HNEAMIP,6HMAXLTR,6HOUTBDS/
COMMON/SECO/ITR

C INITIALIZE NS AND PI

```
      NTC=1
      NS=0
      NPL2=L2-2
      PI(1)=P1I
      PI(2)=P2I
      DO 5 J=3,NP
5      PI(J)=A(J)
C
C      CALL CONV TO ATTEMPT CONVERGENCE TO A SOLUTION USING CURRENT
C      VALUE OF PI AS INITIAL GUESS
C
10      CALL CONV(ICM,PI,NP,DELP,NTC,EPS,ICONV,PF,CSF)
C
C      WRITE RESULTS OF CALL TO CONV
C
      IF(IPRI.NE.1) GO TO 11
      WRITE(6,1006)(PI(KK),KK=1,NP)
      WRITE(6,1007) TYCO(ICONV),ITR,(PF(KK),KK=1,NP)
C
C      IF THERE IS CONVERGENCE TO A NEW SOLUTION STORE, IF NOT STEP
C      UP PARAMETERS
C
11      IF(ICONV.NE.1) GO TO 40
C
C      ...IS THIS FIRST SOLUTION
C
      IF(NS.EQ.0)GO TO 23
C
C      ...TEST FOR NEW SOLUTION
C
      DO 20 I=1,NS
      DO 15 J=3,NP
      IF(ABS(SOLP(J,I)-PF(J)).GT.0.25*DELP(J)) GO TO 20
15      CONTINUE
      GO TO 40
20      CONTINUE
C
C      ...THIS IS NEW SOLUTION THEREFOR STORE
C
23      NS=NS+1
      IE(NS.GT.MS)GO TO 60
      DO 25 J=1,NP
25      SOLP(J,NS)=PF(J)
      DO 30 J=1,NCS
30      SOLC(J,NS)=CSE(J)
C
C      ...STEP UP PARAMETERS
C
40      DO 50 J=1,NPL2
      JJ=NP+1-J
      PI(JJ)=PI(JJ)+DELP(JJ)
      IF(PI(JJ).LT.(B(JJ)+0.01*DELP(JJ)))GO TO 10
50      PI(JJ)=A(JJ)
      RETURN
C
C      WRITE MAXIMUM NUMBER OF SOLUTIONS MESSAGE
C
```

```
60      WRITE(6,1001)(PI(JJ),JJ=1,NP)
      RETURN
```

```
C
C      FORMAT STATEMENTS
C
```

```
1001      FORMAT(1H0,'***** NUMBER OF SOLUTIONS EXCEEDS ALLOTTED '
      'STORAGE. LAST SEARCH POINT WAS',/7X,10E12.5)
1004      FORMAT(1H,'NO. ITR. = ',I3,2X,'PF = ',(T23,6E15.8))
1006      FORMAT(1H,'PI = ',(T23,6E15.8))
1007      FORMAT(1H,'A6,I3,' ITRS PF = ',(T23,6E15.8))
      END
```

APR26, SEEK3

C TITLE SEEK3 - TOTAL DOMAIN SEARCH

C AUTHOR P.F. LONG

C DATE 1/3/69

C PURPOSE TO FIND ALL DISTINCT SOLUTIONS WITHIN THE GIVEN
PARAMETER SPACE.

C METHOD OPTIMUM DOMAIN SEARCH

C CALL SEEK3(A,B,DELP,P1I,P2I,NP,NCS,ICM,EPS,NS,SOL,SOLC,IPRI)

C INPUT THROUGH ARGUMENT LIST

C NAME DEFINITION

C A(I) LOWER BOUND ON ITH INDEPENDENT PARAMETER

C B(I) UPPER BOUND ON ITH INDEPENDENT PARAMETER

C DELP(I) STEP SIZE FOR ITH INDEPENDENT PARAMETER

C P1I VALUE OF P(1) WHILE SEARCHING

C P2I VALUE OF P(2) WHILE SEARCHING

C NP NUMBER OF INDEPENDENT PARAMETERS

C NCS NUMBER OF EQUALITY CONSTRAINTS

C ICM ITERATIVE METHOD SELECTOR

C EPS(I) TOLERANCE ON ITH EQUALITY CONSTRAINT

C IPRI PRINT CONTROL

C OUTPUT THROUGH ARGUMENT LIST

C NAME DEFINITION

C NS NUMBER OF DISTINCT SOLUTIONS FOUND

C SOLP(I,J) ITH COMPONENT OF JTH SOLUTION

C SOLC(I,J) FINAL VALUE OF ITH CONSTRAINT OF JTH SOLUTION

C SUBROUTINES CONSTR, CONV3

C SUBROUTINE STATEMENT

SUBROUTINE SEEK3(A,B,DELP,P1I,P2I,NP,NCS,ICM,EPS,NS,
SOLP,SOLC,IPRI)

C SPECIFICATION STATEMENTS

PARAMETER MS=10000

INCLUDE PARSTA,LIST

DIMENSION A(1),B(1),DELP(1),EPS(1),SOLP(MP,MS),

SOLC(MC,MS),PI(MP),PF(MP),CSF(MC),NGP(MP),

TYCO(5),E(MSP),DEL1(MP)

DATA TYCO/6HCONVER,6HNEASOL,6HNEAMIP,6HMAXITR,6HOUTBDS/
COMMON/SECO/ITR

LOGICAL CLOSE,VCP

C INITIALIZE


```

NTIC=0
IU=34
ITRY=0
IPER=10
NS=0
NPL2=NP-2
PI(1)=P1I
PI(2)=P2I
NPT=1
DO 4 J=1, NP
DEL1(J)=1.01*DELP(J)
DO 5 J=3, NP
NGP(J)=(B(J)-A(J))/DELP(J)+1.01
NPT=NPT*NGP(J)
PI(J)=A(J)
IF(NPT.GE.*MSP) GO TO 100
NST=(IPER*NPT)/100
NU=0
RE=IND IU

```

C
C COMPUTE SUM OF SQUARES FOR EACH GRID POINT

```

C
10 CALL CONSTR(PI,CSE)
NU=NU+1
E(NU)=0.
DO 15 J=1,NCS
15 E(NU)=E(NU)+CSE(J)**2

```

C
C WRITE E, P, AND CSE

```

C
IF (IPRI.NE.1) GO TO 19
WRITE(6,1002)E(NU),(CSE(J),J=1,NCS)
WRITE(6,1003)(PI(J),J=3,NP)

```

C
C STORE X AND E

```

C
19 WRITE(IU) (PI(J+2),J=1,NCS),(CSE(J),J=1,NCS)

```

C
C STEP UP PARAMETERS

```

C
DO 75 J=1,NPL2
JJ=NP+1-J
PI(JJ)=PI(JJ)+DELP(JJ)
IF(PI(JJ).LT.(B(JJ)+0.01*DELP(JJ))) GO TO 10
75 PI(JJ)=A(JJ)

```

C
C FIND THE NEXT SMALLEST E

```

C
NCC=0
35 IL=1
DO 40 J=2,NPT
IF(E(J).LT.E(IL)) IL=J
40 CONTINUE
E(IL)=1.E38

```

C
C FIND GRID POINT IL ON THE DRUM

C

```

      IW=IL-1
      REWIND IU
      IF (IW.EQ. 0) GO TO 49
      DO 48 I=1, IW
48      READ(IU) 0
49      READ(IU) (PI(J+2), J=1, NCS), (CSF(J), J=1, NCS)
      NCC=NCC+1
C
C      TEST FOR NEAR SOLUTION
C
      CLOSE=VCP(SOLP, PI, DEL1, NS, NP)
      IF(CLOSE) GO TO 35
C
C      CALL CONV3
C
      CALL CONV(ICM, PI, NP, DELP, NTC, EPS, ICONV, PF, CSF)
C
C      WRITE RESULTS OF CALL TO CONV
C
      IF(IPRI.NE.1) GO TO 51
      WRITE(6,1006) (PI(KK), KK=1, NP)
      WRITE(6,1007) TYCO(ICONV), ITR, (PF(KK), KK=1, NP)
51      GO TO(23, 65, 70, 70, 70), ICONV
C
C      STORE SOLUTION
C
23      NS=NS+1
      IF(NS.GT.MS) GO TO 60
      DO 25 J=1, NP
25      SOLP(J, NS)=PF(J)
      DO 30 J=1, NCS
30      SOLC(J, NS)=CSF(J)
C
C      RESET NUMBER OF TRYS FLAG
C
65      ITRY=0
      NST=(IPER*(NPT-NCC))/100
      GO TO 35
C
C      UPDATE ITRY
C
70      ITRY=ITRY+1
      IF(ITRY.LT.NST.AND.NCC.LT.NPT) GO TO 35
      REWIND IU
      RETURN
C
C      WRITE MAXIMUM NUMBER OF SOLUTIONS MESSAGE
C
60      WRITE(6,1001) (PI(JJ), JJ=1, NP)
      REWIND IU
      RETURN
100     WRITE(6,1004) NPT
      REWIND IU
      RETURN
C
C      FORMAT STATEMENTS
C
```

```
1001      FORMAT(1H0,'***** NUMBER OF SOLUTIONS EXCEEDS ALLOTTED '  
          'STORAGE. LAST SEARCH POINT WAS'/7X,10E12.5)  
1002      FORMAT(1H,'E = ',E15.8,4X,'E = ',(T24,6E15.8))  
1003      FORMAT(1H,23X,'X = ',(T28,6E15.8))  
1004      FORMAT(1H,'*****',I7,' TOTAL NUMBER OF GRID POINTS '  
          'EXCEEDS ALLOTTED STORAGE *****')  
1005      FORMAT(1H,'***** NTRAM ERROR STOP *****')  
1006      FORMAT(1H,16X,'PI = ',(T23,6E15.8))  
1007      FORMAT(1H,A6,I3,' ITRS PF = ',(T23,6E15.8))  
          END
```

APR26, CONV

C TITLE CONV - ITERATIVE METHOD SELECTOR

C AUTHOR P.F. LONG

C DATE 2-12-68

C PURPOSE TO DIRECT CONTROLE TO ITERATIVE METHOD INDICATED BY ICM

C CALL CONV(ICM,PI,NP,DELP,NCS,EPS,ICONV,PF,CSF)

C INPUT NAME DEFINITION

C ICM INDICATES ITERATIVE METHOD TO BE USED

C PI(J) INITIAL INDEPENDENT PARAMETER VECTOR

C NP NUMBER OF INDEPENDENT PARAMETERS

C DELP(J) STEP SIZE FOR JTH PARAMETER

C NCS NUMBER OF EQUALITY CONSTRAINTS

C EPS(J) TOLERANCE ON JTH CONSTRAINT

C OUTPUT NAME DEFINITION

C ICONV =1 ITERATIVE METHOD CONVERGED, =0 DID NOT

C PF(J) SOLUTION VECTOR IF ICONV = 1, LAST ITERATION

C PARAMETER VALUES IF ICONV = 0

C CSF(J) FINAL VALUE OF JTH CONSTRAINT

C SUBROUTINES CONV1, CONV2, CONV3, CONV4, CONV5

C SUBROUTINE STATEMENT

SUBROUTINE CONV(ICM,PI,NP,DELP,NCS,EPS,ICONV,PF,CSF)

C SPECIFICATION STATEMENTS

DIMENSION PI(1),DELP(1),EPS(1),PF(1),CSF(1)

DIRECT CONTROLE TO ITERATIVE METHOD INDICATED BY ICM

GO TO(5,10,15,20,25),ICM

5 CALL CONV1(PI,NP,DELP,NCS,EPS,ICONV,PF,CSF)

GO TO 30

10 CALL CONV2(PI,NP,DELP,NCS,EPS,ICONV,PF,CSF)

GO TO 30

15 CALL CONV3(PI,NP,DELP,NCS,EPS,ICONV,PF,CSF)

GO TO 30

20 CALL CONV4(PI,NP,DELP,NCS,EPS,ICONV,PF,CSF)

GO TO 30

25 CALL CONV5(PI,NP,DELP,NCS,EPS,ICONV,PF,CSF)

30 RETURN

END

APR26.VCP

C TITLE VCP - VECTOR COMPARE

C AUTHOR P. F. LONG

C DATE 11/22/68

C PURPOSE TO DETERMINE IF A VECTOR IS SUFFICIENTLY CLOSE TO ANY
C ONE OF A SET OF VECTORS

C METHOD COMPARE VECTORS COMPONENT BY COMPONENT

C INPUT BY ARGUMENT LIST

C VSET(I,J) ITH COMPONENT OF JTH VECTOR

C VEC(I) ITH COMPONENT OF VECTOR TO BE COMPARED

C DEL(I) DEFINES SUFFICIENTLY CLOSE

C NV NUMBER OF VECTORS IN THE SET

C NC NUMBER OF COMPONENTS

C OUTPUT VEC = T THE GIVEN VECTOR IS SUFFICIENTLY CLOSE TO
C AT LEAST ONE OF THE SET

C VEC = F THE CONTRARY IS TRUE

C FUNCTION STATEMENT

C LOGICAL FUNCTION VCP(VSET,VEC,DEL,NV,NC)

C SPECIFICATION STATEMENT

C INCLUDE PARSTA,LIST

C DIMENSION VSET(MP,MS),VEC(NC),DEL(NC)

C INITIALIZE

C VCP=.FALSE.
C IF(NV.EQ.0) RETURN

C COMPARE VECTOR WITH SET

```

C      DO 20 I=1,NV
C      DO 10 J=1,NC
C      IF(ABS(VSET(J,I)-VEC(J)).GT.DEL(J))GO TO 20
10      CONTINUE
C      VCP=.TRUE.
C      RETURN
20      CONTINUE
C      RETURN
C      END

```

CONV3

C TITLE CONV3 - SOLUTION FOR A SYSTEM OF NON-LINEAR EQUATIONS
C
C AUTHOR W.B. GRAGG AND J.M. ORTEGA (REVISED BY P.F. LONG)
C
C DATE 2-15-68
C
C PURPOSE TO SOLVE A SYSTEM OF N NON-LINEAR EQUATIONS IN N UNKNOWNNS
C
C METHOD THE N DIMENSIONAL SECANT METHOD
C
C INPUT THROUGH ARGUMENT LIST

NAME	DEFINITION
PI(J)	PI(1)=P1,PI(2)=P2 PI(J),J=3,NP IS THE STARTING POINT FOR THE ITERATIVE PROCEDURE
NP	NUMBER OF PARAMETERS (N=NP-2)
DELP(J)	STEP SIZE FOR JTH INDEPENDENT PARAMETER
NTC	IF NTC.NE.1, Y IS ASSUMED TO BE THE RESULTS OF CALL CONSTR(PI,Y). IF NTC.EQ.1 THE CALL WILL BE MADE.
EPS(J)	TOLERANCE ON JTH CONSTRAINT
CSF(J)	VALUE OF OF SYSTEM AT P(3),...,P(NP)

C OUTPUT THROUGH ARGUMENT LIST

NAME	DEFINITION
ICONV	=1, METHOD CONVERGED =2, NEAR OLD SOLUTION =4, IN REPEATING LOOP =5, OUT OF SEARCH AREA
PF(J)	PF(1)=PI(1),PF(2)=PI(2),P(J),J=3,NP SOLUTION OR LAST ITERATION
Y(J)	FINAL VALUE OF EQUALITY CONSTRAINTS

C CALL CONV3(PI,NP,DELP,NTC,EPS,ICONV,PF,Y)

C SUBROUTINES GAUSS,CONSTR

C NOTE CONV3 IS DESIGNED TO INTEREACE WITH PAR. BECAUSE OF THIS
C PI IS A VECTOR WITH NP COMPONENTS. TWO OF THESE ARE THE
C FIXED GRID PARAMETERS AND NP-2 ARE THE INITIAL POINT
C FOR THE SECANT METHOD, I.E., N=NP-2.

C SUBROUTINE STATEMENT

SUBROUTINE CONV3(PI,NP,DELP,NTC,EPS,ICONV,PF,Y)

C SPECIFICATION STATEMENTS

```

INCLUDE PARSTA,LIST
DIMENSION A(20,20),B(20),Y(20),X(MC),EPS(MC),CEN(MC),
Q(MP,MP),W(MP),R(MP),PI(MP),PF(MP),P(MP),
DELP(MP),DEL(MP)

```

```
COMMON/SECO/ITR
INCLUDE KONSOL,LIST
EQUIVALENCE (X(1),P(3))
LOGICAL CLOSE,VCP
```

```
C
C INITIALIZE
```

```
      N=NP-2
      ICONV=5
      BD=1.01
      IF(NTC.EQ.1) BD=2.
      ZINC=.1/FLOAT(N)
      M=N+1
      PF(1)=PI(1)
      PE(2)=PI(2)
      DO 15 I=1,NP
      DEL(1)=BD*DELP(I)
      P(I)=PI(I)
      DO 16 I=1,N
      CEN(I)=X(I)
      L=0
```

WOLF006

WOLF006

WOLF006

```
C
C IF NTC IS 1, COMPUTE INITIAL VALUE OF SYSTEM
```

```
      IF(NTC.EQ.1) CALL CONSTR(P,Y)
```

```
C
C INITIALIZE THE MATRICES Q AND A AND THE VECTOR B
```

```
      DO 2 J=1,M
      DO 1 I=1,N
      Q(I,J)=X(I)
      A(I,J)=Y(I)
      A(M,J)=1.0
      IF(J.EQ.M) GO TO 3
      B(J)=0.0
      X(J)=X(J)+ZINC*DELP(J+2)
      CALL CONSTR(P,Y)
      2 CONTINUE
      3 B(M)=1.0
```

WOLF006

WOLF007

WOLF007

WOLF007

WOLF007

WOLF007

WOLF007

WOLF007

WOLF007

WOLF007

```
C
C START MAIN ITERATION
```

```
      DO 13 I=1,20
```

```
C
C FOR 4 WEIGHTS FOR F
```

```
      DO 5 I=1,N
      W(I)=0.0
      DO 4 J=1,M
      W(I)=W(I)+A(I,J)**2
      5 W(I)=1.0/W(I)
```

```
C
C FORM WEIGHTED RESIDUALS FOR INITIAL GUESSES
```

```
      DO 6 J=1,M
      R(J)=0.0
      DO 6 I=1,N
```

WOLF009

WOLF010

WOLF009

WOLF009

WOLF009

WOLF009

WOLF008

WOLF008

WOLF008

WOLF009

WOLF009

WOLF009

WOLF009

WOLF009

WOLF009

```

6      R(J)=R(J)+W(I)*A(I,J)**2                                WOLF009
C
C      FIND MAXIMUM WEIGHTED RESIDUAL                            WOLF010
C
C      TEMP=0.0                                                  WOLF010
C      DO 7 J=1,M                                                WOLF010
C      IF(R(J).LT.TEMP)GO TO 7                                    WOLF010
C      TEMP=R(J)                                                  WOLF010
C      K=J                                                        WOLF011
7      CONTINUE                                                  WOLF011
C
C      FORM A NEW APPROXIMATION TO X AND EVALUATE F(X)          WOLF011
C
C      CALL GAUSS(M,1,A,B,Y,DET,EPR)                             WOLF011
C      DO 9 I=1,N                                                WOLF011
C      TEMP=0.0                                                  WOLF011
C      DO 8 J=1,M                                                WOLF011
8      TEMP=TEMP+Y(J)*Q(I,J)                                    WOLF011
9      X(I)=TEMP                                                  WOLF012
C
C      IS X OUT OF SEARCH AREA                                  WOLF012
C
C      DO 14 I=1,N                                               WOLF012
C      IF(ABS(CEN(I)-X(I)).GT.BD*DELP(I+2)) GO TO 20
14     CONTINUE
C      IF(NTC.EQ.1) GO TO 17
C
C      IF X IS NEAR A PREVIOUS SOLUTION, RETURN
C
C      CLOSE=VCP(SOLP,P,DEL,NS,NP)
C      IF(CLOSE) ICONV=2
C      IF(CLOSE) GO TO 20
C
C      CALL CONSTR
C
17     CALL CONSTR(P,Y)
C
C      TEST FOR CONVERGENCE                                     WOLF013
C
C      DO 11 I=1,N                                               WOLF013
11     IF(ABS(Y(I)).GT.EPS(I))GO TO 12
C      ICONV=1
C      GO TO 20
C
C      TEST FOR REPEATING LOOP
C
12     DO 26 I=1,N
C      IF(ABS(X(I)-Q(I,K)).GT.1E-9) GO TO 25
26     CONTINUE
C      ICONV=4
C      GO TO 20
C
C      GET READY FOR NEXT ITERATION                             WOLF013
C
C      R(K)=0.0
25     DO 13 I=1,N
C      Q(I,K)=X(I)
C
C      WOLF014
C      WOLF014
```


13 A(I,K)=Y(I)

WOLF01

ICONV=4

20 DO 21 J=1,N

21 PF(J+2)=X(J)

ITR=L

RETURN

END

WOLF01

F. APR 26. GAUSS

C	TITLE	GAUSS	GAUSS001
C			GAUSS002
C	AUTHOR	J.M. ORTEGA	GAUSS003
C			GAUSS004
C	DATE	FEBRUARY 1963	GAUSS005
C			GAUSS006
C	PURPOSE	TO SOLVE THE MATRIX EQUATION $AX=B$	GAUSS007
C			GAUSS008
C	CALL	GAUSS(N,M,A,B,X,DET,ERR)	GAUSS009
C			GAUSS010
C	INPUT	N=DIMENSION OF THE SQUARE MATRIX A (N.LE.20)	GAUSS011
C		M=NUMBER OF COLUMNS OF B (M.LE.20)	GAUSS012
C		A=THE MATRIX A	GAUSS013
C		B=THE MATRIX B	GAUSS014
C			GAUSS015
C	OUTPUT	X=NUMERICAL SOLUTION OF $AX=B$	GAUSS016
C		DET=DETERMINANT OF A	GAUSS017
C		ERR=AN INDICATOR OF THE ACCURACY OF X	GAUSS018
C			GAUSS019
C	METHOD	GAUSSIAN ELIMINATION WITH ROW INTERCHANGES TO POSITION	GAUSS020
C		THE MAXIMUM PIVOTAL ELEMENTS. A FIRST PASS OBTAINS AN	GAUSS021
C		APPROXIMATION X_0 TO $A^{*}(-1)*B$. THE RESIDUAL $R=B-A*X_0$ IS	GAUSS022
C		THEN COMPUTED IN DOUBLE PRECISION. A SECOND PASS SOLVES	GAUSS023
C		$A*DX_0=R$ FOR DX_0 . THEN $X=X_0+DX_0$. THE QUANTITY ERR IS AN	GAUSS024
C		INDICATION OF THE CONDITION OF A. IT IS DEFINED AS	GAUSS025
C		$\max(\text{ABS}(DX_0(I,J)/X(I,J)))$ FOR $X(I,J) \neq 0.0$. ERR IS	GAUSS026
C		ROUGHLY THE MAXIMUM RELATIVE ERROR IN X. A BYPRODUCT OF	GAUSS027
C		THE REDUCTION OF A IS ITS DETERMINANT.	GAUSS028
C			GAUSS029
C			GAUSS030
C			GAUSS031
C			GAUSS032
C			GAUSS033
C			GAUSS034

```

SUBROUTINE GAUSS(N,M,A,B,X,DET,ERR)
DIMENSION A(20,20),AA(20,20),B(20,20),BB(20,20),X(20,20),
          XX(20,20),IN(20)

```

```

DO 2 I=1,N
DO 1 J=1,N
AA(I,J)=A(I,J)
DO 2 J=1,M
BB(I,J)=B(I,J)

```

PERFORM THE TRIANGULARIZATION OF A _____ GAUSSOID

NN=N-1	GAUS00F
DO 6 L=1,NN	GAUS00F
IN(L)=0	GAUS00F
KP=0	GAUS00F
T=0.0	GAUS00F
DO 3 K=L,N	GAUS00F
TI=ABS(AA(K,L))	GAUS00F
IF(T.GT.TT)GO TO 3	GAUS00F
T=TI	GAUS00F
KP=K	GAUS00F
CONTINUE	GAUS00F

Line	Code	Statement	Label
		IF(L.GE.KP)GO TO 5	GAUS00
		IN(L)=KP	GAUS00
		DO 4 J=L,N	GAUS00
		T=AA(L,J)	GAUS00
		AA(L,J)=AA(KP,J)	GAUS00
4		AA(KP,J)=T	GAUS00
5		LL=L+1	GAUS00
		DO 6 K=LL,N	GAUS00
		R=AA(K,L)/AA(L,L)	GAUS00
		AA(K,L)=R	GAUS00
		DO 6 J=LL,N	GAUS00
6		AA(K,J)=AA(K,J)-R*AA(L,J)	GAUS00
C			GAUS00
C		THIS COMPLETES THE REDUCTION OF A	GAUS00
C			GAUS00
		DET=AA(N,N)*AA(N-1,N-1)	GAUS00
		J=N-2	GAUS00
		DO 7 I=1,J	GAUS00
7		DET=DET*AA(I,I)	GAUS00
		DO 8 I=1,NN	GAUS00
		IF(IN(I).NE.0)DET=-DET	GAUS00
8		CONTINUE	GAUS00
		DO 19 IT=1,2	GAUS00
		IF(IT.EQ.1)GO TO 10	GAUS00
C			GAUS00
C		COMPUTE RESIDUALS	GAUS00
C			GAUS00
		DO 9 J=1,M	GAUS00
		DO 9 I=1,N	GAUS00
		BB(I,J)=-B(I,J)	GAUS00
9		CALL INPR(A(I,1),X(1,J),20,N,BB(I,J))	GAUS00
C			GAUS00
C		REDUCE RIGHT HAND SIDE	GAUS00
C			GAUS00
10		DO 13 L=1,NN	GAUS00
		KP=IN(L)	GAUS00
		IF(KP.EQ.0)GO TO 12	GAUS00
		DO 11 J=1,M	GAUS00
		T=BB(KP,J)	GAUS00
		BB(KP,J)=BB(L,J)	GAUS00
11		BB(L,J)=T	GAUS01
12		LL=L+1	GAUS01
		DO 13 K=LL,N	GAUS01
		R=AA(K,L)	GAUS01
		DO 13 J=1,M	GAUS01
13		BB(K,J)=BB(K,J)-R*BB(L,J)	GAUS01
C			GAUS01
C		START BACK SUBSTITUTION	GAUS01
C			GAUS01
		DO 15 I=1,M	GAUS01
		XX(N,I)=BB(N,I)/AA(N,N)	GAUS01
		DO 15 L=2,N	GAUS01
		LL=N+1-L	GAUS01
		K=LL+1	GAUS01
		T=0.0	GAUS01
		DO 14 J=K,N	GAUS01
14		T=T+AA(LL,J)*XX(J,I)	GAUS01

```
15      XX(LL,I)=(BB(LL,I)-T)/AA(LL,LL)      GAUS011
C      GAUS011
C      ADD CORRECTIONS.      GAUS011
C      GAUS012
      ERR=0.0      GAUS012
      IF(IT.GT.1)GO TO 17      GAUS012
      DO 16 I=1,N      GAUS012
      DO 16 J=1,M      GAUS012
16      X(I,J)=XX(I,J)      GAUS012
      GO TO 19      GAUS012
17      DO 18 I=1,N      GAUS012
      DO 18 J=1,M      GAUS012
      T=X(I,J)      GAUS012
      TT=XX(I,J)      GAUS013
      IF(ABS(T).LT.1E-37) GO TO 18
      TIT=ABS(TT/T)      GAUS013
      IF(ERR.LT.TIT)ERR=TIT      GAUS013
18      X(I,J)=T-TT      GAUS013
19      CONTINUE      GAUS013
      RETURN      GAUS013
      END      GAUS013
```

APR26.INPR

C

C TITLE -INPR-

C

C AUTHOR REWRITTEN BY M. TRIMBLE, JULY, 1967, FROM MAP TO FORTRAN

C

C PURPOSE TO COMPUTE THE INNER PRODUCT OF VECTORS A AND B OF LENGTH
C J IN DOUBLE PRECISION AND ADD THE SINGLE PRECISION RESULT
C TO C

C

C CALL INPR(A,B,I,J,C)

C

C WHERE

C A = LOCATION OF FIRST WORD OF FIRST VECTOR ARRAY

C B = LOCATION OF FIRST WORD OF SECOND VECTOR ARRAY

C I = 1 IF A IS A ONE DIMENSIONAL ARRAY

C = FIRST DIMENSIONAL SUBSCRIPT OF A IF ARRAY A

C IS TWO DIMENSIONAL

C J = THE NUMBER OF TERMS TO BE TAKEN FOR THE INNER
C PRODUCT (USUALLY THE FIRST DIMENSIONAL

C SUBSCRIPT OF B WHICH SHOULD EQUAL THE SECOND

C DIMENSIONAL SUBSCRIPT OF A)

C C = LOCATION TO WHICH THE RESULT OF COMPUTATION

C IS TO BE ADDED

C

C

SUBROUTINE INPR(A,B,I,J,C)

C

C SPECIFICATION STATEMENTS

C

DIMENSION A(1),B(1)

DOUBLE PRECISION HA,HB,HOLD,HC

C

C INITIALIZATION

C

HC = 0.0

M = I*J

IJ = 0

C

C DO INNER PRODUCT OF VECTORS

C

DO 100 II = 1,M,I

IJ = IJ+1

HA = A(II)

HB = B(IJ)

HOLD = HA*HB

100 HC = HC+HOLD

C

C ADD RESULT TO PREVIOUS RESULT

C

C = HC+C

RETURN

END

APR26.GRIDMK

C TITLE GRIDMK - SOLUTION MAPPING ROUTINE

C AUTHOR MARILYN R. TRIMBLE

C SPONSOR A. A. VANDERVEEN

C DATE 12-1-67

C PURPOSE TO MAP A GIVEN SOLUTION TO ALL POINTS OF A GRID.

C METHOD SOLUTIONS ARE FOUND FOR THE GRID POINTS NEAR THE GIVEN
SOLUTION. THE EXTRAPOLATION OF THE SOLUTIONS FOUND AT
TWO ADJACENT POINTS IS USED AS AN INITIAL GUESS FOR AN
ITERATIVE PROCEDURE IN AN EFFORT TO CONVERGE TO SOLUTIONS
FOR THE POINTS ON EITHER SIDE OF THE ADJACENT SET. IN
THE ABSENCE OF ADJACENT SETS, SOLUTIONS AT THE FOUR
NEIGHBORING POINTS ARE USED AS INITIAL GUESSES.

C CALL CALL GRIDMK(A,B,DELP,EPS,PINI,NP,NCS,ICM,GRD,COPT,IDCV,
IPRE).

C INPUT THROUGH CALL LIST

C A(J) MINIMUM VALUE FOR JTH PARAMETER
C B(J) MAXIMUM VALUE FOR JTH PARAMETER
C DELP(J) DELP(1) AND DELP(2) DETERMINE THE GRID MESH
C EPS(J) TOLERANCE ON JTH CONSTRAINT FOR CONV
C PINI(J) INITIAL SOLUTION VALUE FOR JTH PARAMETER
C NP NUMBER OF PARAMETERS
C NCS NUMBER OF EQUALITY CONSTRAINTS
C ICM INDICATOR FOR ITERATIVE METHOD TO BE USED BY CONV

C OUTPUT THROUGH THE CALL LIST

C GRD(I,J,K) RESULTING SET OF SOLUTIONS. I=1,NP J=1,NP1
K=1,NP2 WHERE NP1 AND NP2 ARE THE NUMBER OF
P(1)'S AND P(2)'S RESPECTIVELY
C COPT(J,K) OPTIMUM VALUE WHEN OPTIMIZING
C IDCV(J,K) CONVERGENCE INDICATOR
=1 CONVERGED
=0 DID NOT CONVERGE

C SUBROUTINES CONV SUBROUTINE TO BE USED TO FIND CONVERGENCE

C SUBROUTINE STATEMENT

SUBROUTINE GRIDMK(A,B,DELP,EPS,PINI,NP,NCS,ICM,GRD,COPT,
IDCV,IPRE)

C SPECIFICATION STATEMENTS

C INCLUDE PARSTA,LIST

C DIMENSION PINI(1),A(1),B(1),DELP(1),GRD(MP,MA,MO),
UHQLD(2),DELD(4),DELY(4),UCOMP(2),USTART(2),UBASE(2),
NLOOP(4),NLP(4),NDIM(2),PI(MP),PE(MP),IDCV(MA,MO),

```
      COPT(MA,MO),CSE(KC),XX(1000),AN(2),BX(2),YY(1000)
      DIMENSION TYCO(5)
      DATA TYCO/6HCONVER,6HNEASOL,6HNEAMIP,6HMAXITR,6HOUTBDS/
      COMMON/SECO/ITR
```

```
C
C  INITIALIZE PROGRAM
C
```

```
      IGR=1
      PER=0.0
      ISTART=0
      DO 10 I=1,2
10      NDIM(I)=(B(I)-A(I))/DELP(I)+1.01
      TGP=NDIM(2)*NDIM(1)
      CGP=0.0
```

```
C
      I1=NDIM(1)
      I2=NDIM(2)
      DO 15 J=1,I2
      DO 15 I=1,I1
15      IDCV(I,J)=0
```

```
C
C  SET MINIMUM AND MAXIMUM VALUES FOR SEARCH
C
```

```
      AN(1) = A(1)-DELP(1)/10.0
      AN(2) = A(2)-DELP(2)/10.0
      BX(1) = B(1)+DELP(1)/10.0
      BX(2) = B(2)+DELP(2)/10.0
```

```
C
C  SET INCREMENTS FOR A TWO DIMENSIONAL GRID
C
```

```
100      DELX(1)=DELP(1)
      DELX(2)=DELP(1)
      DELX(3)=-DELP(1)
      DELX(4)=-DELP(1)
      DELY(1)=DELP(2)
      DELY(2)=-DELP(2)
      DELY(3)=-DELP(2)
      DELY(4)=DELP(2)
```

```
C
C  FIND INITIAL POINT ON THE GRID
C
```

```
      DO 125 I=1,2
      UHOLD(I)=A(I)
115      UHOLD(I)=UHOLD(I)+DELP(I)
      IF (UHOLD(I).GT.PINI(I)) GO TO 125
      GO TO 115
125      UHOLD(I)=UHOLD(I)-DELP(I)
      DO 135 I=3,MP
135      PI(I)=PINI(I)
```

```
C
C  ATTEMPT TO FIND A SOLUTION AT FOUR SURROUNDING POINTS AND CHOOSE
C  A STARTING POINT AS BASE
C
```

```
      IND=1
145      IF (UHOLD(1).LT.AN(1).OR.UHOLD(1).GT.BX(1)) GO TO 155
      IF (UHOLD(2).LT.AN(2).OR.UHOLD(2).GT.BX(2)) GO TO 155
      IS1=(UHOLD(1)-A(1))/DELP(1)+1.1
```

```

IS2=(UHOLD(2)-A(2))/DELP(2)+1.1
DO 147 I=1,2
147 PI(I)=UHOLD(I)
CALL CONV(ICONV,PI,NP,DELP,IGR,EPS,ICONV,PF,CSF)
C
C WRITE RESULTS OF CALL TO CONV
C
IF(IPRF.NE.1) GO TO 148
WRITE(6,1006) (PI(KK),KK=1,NP)
WRITE(6,1007) TYCO(ICONV),ITR,(PF(KK),KK=1,NP)
143 CONT(IS1,IS2)=CSF(1)
IF(ICONV.NE.1) GO TO 155
CGP=CGP+1.0
DO 150 I=1,NP
150 GRD(I,IS1,IS2)=PF(I)
IDCV(IS1,IS2)=1
IF(ISTART.NE.0) GO TO 155
USTART(1)=UHOLD(1)
USTART(2)=UHOLD(2)
ISTART=1
155 GO TO (160,162,164,170),IND
160 UHOLD(1)=UHOLD(1)+DELP(1)
IND=2
GO TO 145
162 UHOLD(1)=UHOLD(1)+DELP(1)
UHOLD(2)=UHOLD(2)+DELP(2)
IND=3
GO TO 145
164 UHOLD(1)=UHOLD(1)+DELP(1)
IND=4
GO TO 145
170 IF(ISTART.EQ.0) GO TO 750
200 UBASE(1)=USTART(1)
UBASE(2)=USTART(2)
C
C SET UP INDEXING TO COVER ALL POINTS OF GRID FROM INITIAL BASE
C POINT
C
210 DO 215 I=1,4
215 NLP(I)=1
DO 280 KL=0,1000
NEND=0
DO 280 I=1,4
280 NLOOP(I)=0
DO 500 JL=1,4
C
C IF PRESENT BASE SIDE HAS NO BASE POINTS WITHIN LIMITS OF GRID GO
C TO FIRST BASE POINT OF NEXT BASE SIDE
C
IF(NLP(JL).NE.0) GO TO 300
UBASE(1)=UBASE(1)+KL*DELP(JL)
UBASE(2)=UBASE(2)+KL*DELP(JL)
GO TO 500
300 DO 450 IL=1,KL
C
C DETERMINE IF PRESENT BASE POINT HAS A SOLUTION
C

```



```

      IF(UBASE(1).LT.AN(1).OR.UBASE(1).GT.BX(1)) GO TO 420
      IF(UBASE(2).LT.AN(2).OR.UBASE(2).GT.BX(2)) GO TO 420
      IJ1=(UBASE(1)-A(1))/DELP(1)+1.1
      IJ2=(UBASE(2)-A(2))/DELP(2)+1.1
      NLOOP(JL)=1
      IF(IDC1(IJ1,IJ2).EQ.0) GO TO 420
310    NEND=1
C
C    FIND FIRST OF FOUR POINTS SURROUNDING BASE POINT AND ITS OPPOSITE
C
      UHOLD(1)=UBASE(1)-DELP(1)
      UHOLD(2)=UBASE(2)
      UCOMP(1)=UBASE(1)+DELP(1)
      UCOMP(2)=UBASE(2)
C
C    ATTEMPT TO FIND SOLUTION IF POINT IS WITHIN LIMITS OF GRID AND IF
C    POINT DOES NOT ALREADY HAVE SOLUTION
C
      DO 400 J=1,4
      IF(UHOLD(1).LT.AN(1).OR.UHOLD(1).GT.BX(1)) GO TO 370
      IF(UHOLD(2).LT.AN(2).OR.UHOLD(2).GT.BX(2)) GO TO 370
      IS1=(UHOLD(1)-A(1))/DELP(1)+1.1
      IS2=(UHOLD(2)-A(2))/DELP(2)+1.1
      IF(IDC1(IS1,IS2).EQ.1) GO TO 370
C
C    DETERMINE INITIAL GUESS FOR SOLUTION FROM BASE POINT SOLUTION AND
C    FROM OPPOSITE POINT SOLUTION IF PRESENT
C
      DO 333 I=1,2
333    PI(I)=UHOLD(I)
      IF(UCOMP(1).LT.AN(1).OR.UCOMP(1).GT.BX(1)) GO TO 335
      IF(UCOMP(2).LT.AN(2).OR.UCOMP(2).GT.BX(2)) GO TO 335
      IC1=(UCOMP(1)-A(1))/DELP(1)+1.1
      IC2=(UCOMP(2)-A(2))/DELP(2)+1.1
      IF(IDC1(IC1,IC2).EQ.1) GO TO 345
335    DO 340 I=3,NP
340    PI(I)=GRD(I,IJ1,IJ2)
      GO TO 350
345    DO 346 I=3,NP
346    PI(I)=2.0*GRD(I,IJ1,IJ2)-GRD(I,IC1,IC2)
C
C    CALL CONV FOR SOLUTION
C
350    IF(IPRF.NE.1) GO TO 351
      WRITE(6,1006) (PI(KK),KK=1,NP)
351    CALL CONV(ICM,PI,NP,DELP,IGR,EPS,ICONV,PF,CSF)
C
C    WRITE RESULTS OF CALL TO CONV
C
      IF(IPRF.NE.1) GO TO 355
      WRITE(6,1007) TYCO(ICONV),ITR,(PF(KK),KK=1,NP)
355    IF(ICONV.NE.1) GO TO 370
      CGP=CGP+1.0
      DO 360 I=1,NP
360    GRD(I,IS1,IS2)=PE(I)
      IDC1(IS1,IS2)=1
C

```

C CALCULATE NEXT POINT NEAR PRESENT BASE POINT

C

370 UHOLD(1)=UHOLD(1)+DELX(J)

UHOLD(2)=UHOLD(2)+DELY(J)

UCOMP(1)=UCOMP(1)-DELX(J)

UCOMP(2)=UCOMP(2)-DELY(J)

400 CONTINUE

C

C CALCULATE NEXT BASE POINT ON PRESENT BASE LINE

C

420 UBASE(1)=UBASE(1)+DELX(JL)

UBASE(2)=UBASE(2)+DELY(JL)

450 CONTINUE

IF(NLOOP(JL).EQ.0) NLP(JL)=0

500 CONTINUE

C

C IF THERE ARE NO REMAINING BASE POINTS WITH A SOLUTION RETURN TO

C

MAIN PROGRAM

C

IF(NEND.EQ.0) GO TO 650

C

C CALCULATE INITIAL BASE POINT FOR NEW BASE LINE

C

UBASE(1)=UBASE(1)-DELP(1)

600 CONTINUE

C

C CALCULATE PERCENTAGE OF TOTAL POINTS THAT CONVERGED TO A SOLUTION

C

650 PERH=PER

PER=(CGP/TGP)*100.0

WRITE(6,1004) PER

IF(PER-99.9) 660,660,700

660 IF(PER-PERH) 700,700,200

700 CONTINUE

C

C PLOT THE GRID POINTS THAT CONVERGED

C

K=0

DO 745 N=1,I1

DO 740 M=1,I2

IF(IDCV(N,M).EQ.0) GO TO 740

730 K=K+1

XX(K)=GRD(1,N,M)

YY(K)=GRD(2,N,M)

740 CONTINUE

745 CONTINUE

K=K+1

XX(K)=A(1)

YY(K)=A(2)

K=K+1

XX(K)=A(1)

YY(K)=B(2)

K=K+1

XX(K)=B(1)

YY(K)=A(2)

K=K+1

XX(K)=B(1)

```
      YY(K)=R(2)  
      CALL RKPL0T(XX,YY,K)  
750    WRITE(6,1005) PER  
      RETURN
```

C

C FORMAT STATEMENTS

C

```
1001    FORMAT(1H ,16X,'PI = ',(T23,6E15.8))  
1004    FORMAT(1H0,'*****PERCENTAGE OF POINTS CONVERGING = ',  
      F7.3,'*****'/)  
1005    FORMAT(1H0,'POINTS FOR WHICH SOLUTIONS WERE FOUND - P1 '  
      'VERTICAL AXIS, P2 HORIZONTAL AXIS, PERCENTAGE ',F7.3///)  
1006    FORMAT(1H ,16X,'PI = ',(T23,6E15.8))  
1007    FORMAT(1H ,A6,I3,' ITRS  PF = ',(T23,6E15.8))
```

END

L APR26.FRMPLT

C TITLE FRMPLT - INTERPOLATION AND PLOTTING CONTROL

C AUTHOR M. T. HINMAN

C PURPOSE TO READ FUNCTIONS FROM FASTDRUM AND CALL INTERPOLATION
AND PLOTTING ROUTINES.

C METHOD FRMPLT READS THE FUNCTION TO BE PLOTTED FROM ITS FASTDRUM
FILE. IF IT HAS NOT PREVIOUSLY BEEN INTERPOLATED, INGRID
IS CALLED. THE CONTOURS ARE PLOTTED AND THEN SAVED IF
THEY ARE TO BE USED AGAIN.

C INPUT VARIABLES FROM COMMON MCL
VARIABLES FROM COMMON IMP
THE FASTDRUM FILES GENERATED BY PLTCON

C SUBROUTINE STATEMENT

C SUBROUTINE FRMPLT

FRMPLT

C SPECIFICATION STATEMENTS

C INCLUDE PARSTA,LIST
INCLUDE KOMMCL,LIST
INCLUDE GRIDSP,LIST
INCLUDE KPEILE,LIST
PARAMETER IMAXSF = 4

FRMPLT

FRMPLT

C IMAXSF IS THE MAXIMUM NUMBER OF SCRATCH UNITS AVAILABLE FOR
INTERMEDIATE STORAGE AT THE PRESENT TIME

C DIMENSION TTLT(8),ATLT(8),OTLT(8),IDCV(MA,MO),F(MA,MO),
HP1(MA),HP2(MO),DELTA(2),NTABSU(10)

FRMPLT

C NTABSU CONTAINS THE NUMBERS OF THE AVAILABLE SCRATCH UNITS

C DATA NTABSU/21,22,23,24,1,2,3,4,2*0/

C SET NUMBER OF SCRATCH UNIT FOR OVERFLOW DURING INTERPOLATION

C NF = 20

C SET VARIABLES FOR CALLING SCIPLOT

C NDEH=1
NCHAR=42
GRIDD=20.
NX=2
NY=2
LX=0
LY=0
NV=1
NS=2
NG=2

FRMPLT

```

C      SET INITIAL CONDITIONS
C
C      III = NA
C      IPAC = 0
C      IFRM=0
C      NFILE=1
C      IOVFU = 0
C      CALL CAMRAV(935)
C      CALL LSIG(5,5)
C
C      SET LOOP FOR PROCESSING ONE FRAME AT A TIME
C
C      130      IFRM=IFRM+1
C      JFUNC=0
C
C      SET FRAME LABELS
C
C      XL=AMIN(IFRM)
C      XR=AMAX(IFRM)
C      YL=OMIN(IFRM)
C      YU=OMAX(IFRM)
C      DO 150 I=1,8
C      TTLT(I)=TLT(I,IFRM)
C      ATLT(I)=ALA(I,IFRM)
C      150      OTLT(I)=OLA(I,IFRM)
C      NFADV = 1
C
C      SET LOOP FOR ONE FUNCTION AT A TIME
C
C      170      JFUNC = JFUNC+1
C      IDRM=IFD(IFRM,JFUNC)
C
C      IF PLOTTING ARRAY IS ALREADY INTERPOLATED TRANSFER TO READ IN
C
C      IDM = NEDRM(IDRM)
C      IF (IDM.GT.1.OR.IDM.LT.0) GO TO 500
C
C      READ FUNCTION ARRAY FROM DRUM FILE
C
C      READ(IDRM) NT,NHSOL,IVRH,NA,NO
C      DO 200 IT1=1,NA
C      DO 200 IT2=1,NO
C      200      READ(IDRM)IDCV(IT1,IT2),E(IT1,IT2)
C      READ(IDRM) (HP1(I),I=1,NA), (HP2(J),J=1,NO)
C
C      PRINT FUNCTION ARRAY IF REQUESTED
C
C      IF (IPRP.LT.1) GO TO 210
C      IPAC = IPAC+1
C      WRITE(6,2) IPAC,NT,NHSOL,IVRH
C      CALL PRTOUT(NA,NO,HP1,HP2,E)
C
C      SET VARIABLES FOR INTERPOLATION
C
C      210      BP1=HP1(1)
C      BP2=HP2(1)
C      NP1=NA

```

```

      NP2=NO
      DP1=HP1(2)-HP1(1)
      DP2=HP2(2)-HP2(1)
      DELTA(1)=P1INC(IFRM,JFUNC)
      DELTA(2)=P2INC(IFRM,JFUNC)
      IPRT=IPRP-2
C
C      TRANSFER TO VERTICAL CONTOUR SLICING ROUTINE IF REQUESTED
C
      IF(IVS(IFRM,JFUNC).GT.0) GO TO 350
C
C      CALL INTERPOLATION ROUTINE FOR LEVEL SLICING
C
      CMNV = CMIN(IFRM,JFUNC)
      CMXV = CMAX(IFRM,JFUNC)
      CDELV = DELC(IFRM,JFUNC)
      CALL INGRID(BP1,BP2,DP1,DP2,F,NP1,NP2,IDCV,CMNV,CMXV,
      CDELV,DELTA)
      REWIND NF
      REWIND NFUI
      GO TO 400
C
C      CALL INTERPOLATION ROUTINE FOR VERTICAL SLICING
C
350      IDS=IVS(IFRM,JFUNC)
      ISMIN=IS(IFRM,JFUNC)
      ISMAX=LS(IFRM,JFUNC)
      ISINC=INCS(IFRM,JFUNC)
      IF(IDS.EQ.1)DELTA(1)=DELTA(2)
      CALL VSLIC(BP1,BP2,DP1,DP2,F,NP1,NP2,IDCV,IDS,ISMIN,
      ISMAX,ISINC,DELTA)
C
C
400      NFUI = IDRM
      NFUI = NF
      IF(NTOF.EQ.0) GO TO 410
      IE(NEDRM(IDRM).EQ.0) GO TO 410
      IOVFU = IOVFU+1
      IE(IOVFU.GT.IMAXSF) NEDRM(IDRM)=0
      NFUI = NTA3SU(IOVFU)
410      IE(NIPOR(IDRM).EQ.0) GO TO 550
      IF(NTOF.NE.0) GO TO 440
C
C      WRITE HEADING FOR ON-LINE PLOT
C
      IK=IKOUNT
      IE(IVS(IFRM,JFUNC).GT.0) GO TO 420
      XX(IKOUNT+1)=HP1(1)
      YY(IKOUNT+1)=HP2(1)
      XX(IKOUNT+2)=HP1(1)
      YY(IKOUNT+2)=HP2(NO)
      XX(IKOUNT+3)=HP1(NA)
      YY(IKOUNT+3)=HP2(1)
      XX(IKOUNT+4)=HP1(NA)
      YY(IKOUNT+4)=HP2(NO)
      IK = IKOUNT+4
420      CALL BKPLOT(XX,YY,IK)

```

WRITE MESSAGE STATING THAT THERE ARE TOO MANY POINTS TO BE
PLOTTED-ON-LINE

GO TO 550

440 WRITE(6,2) IPAC,NT,NHSOL,IVRH
WRITE(6,3)

GO TO 550

500 IF(NFDRM(IDRM).GT.1) GO TO 505

IN = -NFDRM(IDRM)

NFUI = NTAJSU(IN)

GO TO 510

505 NFUI = IDRM

510 READ(NFUI) IKOUNT,NTOF

READ(NFUI) (XX(I),YY(I),I=1,IKOUNT)

550 IF (NFDRM(IDRM) .NE. 1) GO TO 570

REWIND NFUI

WRITE(NFUI) IKOUNT,NTOF

WRITE(NFUI) (XX(I),YY(I),I=1,IKOUNT)

570 CONTINUE

PLOT

IF(IAO(IFRM,JFUNC).EQ.1) GO TO 580

CALL SCPLLOT(LX,LY,XX,YY,IKOUNT,NDEN,NV,NS,NCHAR,NEADV,

NG,GRIDD,GRIDD,NX,XL,XR,NY,YL,YU,TTLT,ATLT,OTLT)

GO TO 590

580 CALL SCPLLOT(LX,LY,YY,XX,IKOUNT,NDEN,NV,NS,NCHAR,NFADV,

NG,GRIDD,GRIDD,NX,XL,XR,NY,YL,YU,TTLT,ATLT,OTLT)

590 NFADV = 2

IF(NTOF.NE.0) GO TO 600

REWIND NFUI

IF(NFDRM(IDRM).EQ.0) GO TO 700

REWIND NFUI

NFDRM(IDRM) = 2

GO TO 700

600 DO 650 IL = 1,NTOF

READ(NFUI) (XX(I),YY(I),I=1,IMAXC)

IF(NFDRM(IDRM).NE.1) GO TO 620

WRITE(NFUI) (XX(I),YY(I),I=1,IMAXC)

620 CONTINUE

PLOT

IKOUNT = IMAXC

IF(IAO(IFRM,JFUNC).EQ.1) GO TO 630

CALL SCPLLOT(LX,LY,XX,YY,IKOUNT,NDEN,NV,NS,NCHAR,NEADV,

NG,GRIDD,GRIDD,NX,XL,XR,NY,YL,YU,TTLT,ATLT,OTLT)

GO TO 650

630 CALL SCPLLOT(LX,LY,YY,XX,IKOUNT,NDEN,NV,NS,NCHAR,NFADV,

NG,GRIDD,GRIDD,NX,XL,XR,NY,YL,YU,TTLT,ATLT,OTLT)

650 CONTINUE

REWIND NFUI

IF(NFDRM(IDRM).NE.1) GO TO 700

REWIND NFUI

NFDRM(IDRM) = -IOVFU

700 IF(JFUNC.LT.NEN(IFRM)) GO TO 170

IF(IERM.LT.NFR) GO TO 130

CALL EOFTV

RETURN

C

C

FORMAT STATEMENTS

C

2 FORMAT(1H1,/,16H FUNCTION ARRAY, I3,10X,10H FROM UNIT, I3,
11H SOLUTION, I3,15H AND VARIABLE, I3//)

3 FORMAT(1H,10X, 'THERE ARE TOO MANY POINTS FOR ON LINE PLO
TTING.')

END